

UNCLASSIFIED

AD NUMBER

ADB127191

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to DoD only; Software Documentation; MAR 1988. Other requests shall be referred to Air Force Wright Aeronautical Laboratories, ATTN: AFWAL/FIBRA, Wright-Patterson AFB, OH 45433-6553. This document contains export-controlled technical data.

AUTHORITY

WL/DOOS ltr dtd 19 Sep 1991

THIS PAGE IS UNCLASSIFIED

DTIC FILE COPY

L ①

AFWAL-TR-88-3028  
Volume II

AD-B127 191

# AUTOMATED STRUCTURAL OPTIMIZATION SYSTEM (ASTROS)



VOLUME II - USER'S MANUAL

D. J. NEILL

E. H. JOHNSON

Northrop Corporation, Aircraft Division  
Hawthorne, California 90250

D. L. HERENDEEN

Universal Analytics, Inc.  
Playa del Rey, California 90291

April 1988

DTIC  
ELECTE  
NOV 29 1988  
S E D

FINAL REPORT FOR PERIOD JULY 1983 - DECEMBER 1987

Distribution authorized to DOD components only; software documentation, March 1988. Other requests for this document must be referred to AFWAL/FIBRA, WPAFB OH 45433-6553. Requests must include a Statement of Terms and Conditions--Release of Air Force-Owned or Developed Computer Software Packages. (See block 16 of DD Form 1473 herein.)

**WARNING** - This document contains technical data whose export is restricted by the Arms Export Control Act (Title 22, U. S. C., Section 2751, et seq.) or The Export Administration Act of 1979, as amended, Title 50, U. S. C., App. 2401, et seq. Violations of these export laws are subject to severe criminal penalties. Disseminate in accordance with the provisions of AFR 80-34. (Include this statement with any reproduced portion.)

**DESTRUCTION NOTICE** - Destroy by any method that will prevent disclosure of contents or reconstruction of the document.

FLIGHT DYNAMICS LABORATORY  
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES  
AIR FORCE SYSTEMS COMMAND  
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6553

88 11 29 016

## REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>			1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION/AVAILABILITY OF REPORT		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			SEE REVERSE SIDE OF THIS FORM		
4. PERFORMING ORGANIZATION REPORT NUMBER(S)  NOR 88-13			5. MONITORING ORGANIZATION REPORT NUMBER(S)  AFWAL-TR-88-3028, Volume II		
6a. NAME OF PERFORMING ORGANIZATION NORTHROP CORPORATION Aircraft Division		6b. OFFICE SYMBOL (If applicable) 3854/82	7a. NAME OF MONITORING ORGANIZATION Analysis & Optimization Branch (AFWAL/FIBR) Air Force Wright Aeronautical Laboratories		
6c. ADDRESS (City, State, and ZIP Code)  Hawthorne, California 90250-3277			7b. ADDRESS (City, State, and ZIP Code)  Wright-Patterson AFB OH 45433-6553		
8a. NAME OF FUNDING / SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F33615-83-C-3232		
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS		
PROGRAM ELEMENT NO. 62201F		PROJECT NO. 2401	TASK NO. 02	WORK UNIT ACCESSION NO. 57	
11. TITLE (Include Security Classification) AUTOMATED STRUCTURAL OPTIMIZATION SYSTEM (ASTROS) VOLUME II - USER'S MANUAL					
12. PERSONAL AUTHOR(S) Neill, D.J., Johnson, E.H., and Herendeen, D.L.					
13a. TYPE OF REPORT Final		13b. TIME COVERED FROM 7/83 TO 12/87		14. DATE OF REPORT (Year, Month, Day) 1988, APRIL, 07	
15. PAGE COUNT					
16. SUPPLEMENTARY NOTATION Copies of Statement of Terms and Conditions--Release of Air Force-Owned or Developed Computer Software Packages will be furnished upon request to AFWAL/IST, WPAFB OH 45433-6553. Export Control Restrictions.					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	AUTOMATED DESIGN, STRUCTURAL ANALYSIS, MULTIDISCIPLINARY ANALYSIS, FLUTTER ANALYSIS, DYNAMIC ANALYSIS		
01	03	03			
01	01	03			
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
<p>The ASTROS (Automated STRuctural Optimization System) procedure provides a multidisciplinary analysis and design capability for aerospace structures. The engineering analysis capabilities in the system include structural analysis (static and dynamic), aeroelastic analysis (static and dynamic) and automated design. A specifically designed data base and executive system were implemented to maximize the system's efficiency, flexibility, and maintainability.</p> <p>The final report consists of four volumes:</p> <p>VOLUME I - ASTROS Theoretical Manual VOLUME II - ASTROS User's Manual VOLUME III - ASTROS Application Manual VOLUME IV - ASTROS Programmer's Manual</p> <p style="text-align: right;">) over</p>					
(CONTINUED ON REVERSE SIDE OF THIS FORM)					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT <input checked="" type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Capt. R. A. Canfield			22b. TELEPHONE (Include Area Code) (513) 255-6992		22c. OFFICE SYMBOL AFWAL/FIBRA

3. (CONTINUED)

Distribution authorized to DOD components only; software documentation, Mar 88. Other requests for this document shall be referred to AFWAL/FIBRA, WPAFB OH 45433-6553. Requests must include a Statement of Terms and Conditions--Release of Air Force-Owned or Developed Computer Software Packages. (See Block 16 of this form.)

19. (CONTINUED)

This report is the User's Manual for the ASTROS system. As such, it contains descriptions of the input data, which is made up of four optional packets: (1) Debug, (2) Executive System, (3) Solution Control, and (4) Bulk Data packets. ASTROS output is also described in order to permit interpretation of results. Appendices give detailed information on data preparation and on the use of advanced features that permit the user to modify the standard execution of ASTROS.

(KR)



## FOREWORD

Contract F33615-83-C-3232, entitled "Automated Strength-Aeroelastic Design of Aerospace Structures," was initiated by the Analysis and Optimization Branch (FIBR) of the Air Force Wright Aeronautical Laboratories. The objective of this contract was to develop a computer procedure which can assist significantly in the preliminary automated design of aerospace structures. This report, which is one of a four-volume final report, is a User's Manual for the delivered computer procedure.

Northrop Corporation, Aircraft Division, was the primary contractor for this program with Universal Analytics, Inc. (UAI) and Kaman Avidyne acting as subcontractors. The principal contributors to this report were: D. J. Neill, Project Co-Principal investigator, E. H. Johnson, the overall Program Manager at Northrop and D. L. Herendeen, the Project Manager at UAI. R. E. Blauvelt and E. S. Saether at Northrop also contributed to the writing of this report.

Capt. R. A. Canfield was the Air Force Project Manager while Dr. V. B. Venkayya initiated the program at the Air Force and provided overall program direction. The work reported on in this report was performed from 01 July 1983 through 31 December 1987.

<b>Accession For</b>	
NTIS GRA&I	<input type="checkbox"/>
DTIC TAB	<input checked="" type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
<b>Availability Codes</b>	
Dist	Avail and/or Special
E-4	57 JK



## TABLE OF CONTENTS

<u>SECTION</u>		<u>PAGE</u>
I	INTRODUCTION. . . . .	1
II	THE INPUT DATA STREAM . . . . .	5
	2.1 The Assign Database Entry. . . . .	7
	2.2 The Include Directive. . . . .	10
III	THE DEBUG PACKET. . . . .	11
	3.1 Executive System Debug Commands. . . . .	13
	3.2 Data Base and Memory Manager Debug Commands. . . . .	13
IV	THE EXECUTIVE SYSTEM (MAPOL) PACKET . . . . .	15
	4.1 The MAPOL Program. . . . .	17
	4.2 MAPOL Edit Commands. . . . .	17
	4.3 The Standard Executive Sequence. . . . .	18
	4.4 Standard Executive Sequence Structure. . . . .	19
	4.4.1 MAPOL Declarations. . . . .	19
	4.4.2 The Solution Algorithm. . . . .	28
	4.4.3 Modifying the Standard MAPOL Sequence . . . . .	46
V	THE SOLUTION CONTROL PACKET . . . . .	51
	5.1 Type of Boundary Condition . . . . .	52
	5.2 Boundary Condition . . . . .	53
	5.3 Disciplines. . . . .	58
	5.4 Discipline Options . . . . .	60
	5.4.1 STATICS Discipline Options. . . . .	61
	5.4.2 MODES Discipline Options. . . . .	63
	5.4.3 SAERO Discipline Options. . . . .	63
	5.4.4 FLUTTER Discipline Options. . . . .	64
	5.4.5 TRANSIENT Discipline Options. . . . .	64
	5.4.6 FREQUENCY Discipline Options. . . . .	65
	5.4.7 BLAST Discipline Options. . . . .	65
	5.5 Output Requests. . . . .	65
	5.5.1 Subcase Options . . . . .	66
	5.5.2 Response Quantity Options . . . . .	68
	5.5.3 Form Options. . . . .	69
	5.5.4 Labeling Options. . . . .	71

# TABLE OF CONTENTS (Continued)

<u>SECTION</u>		<u>PAGE</u>
VI	THE BULK DATA PACKET. . . . .	73
6.1	Format of the Bulk Data Entry. . . . .	74
6.2	Data Field Formats . . . . .	77
6.3	Error Checking in the Input File Processor . . . . .	78
6.4	Bulk Data Entry Summary. . . . .	78
6.4.1	Aerodynamic Load Transfer. . . . .	79
6.4.2	Applied Dynamic Loads. . . . .	79
6.4.3	Applied Static Loads . . . . .	79
6.4.4	Boundary Condition Constraints . . . . .	79
6.4.5	Design Constraints . . . . .	80
6.4.6	Design Variables, Linking Options and Optimization Parameters. . . . .	80
6.4.7	Geometry . . . . .	80
6.4.8	Material Properties. . . . .	81
6.4.9	Miscellaneous Inputs . . . . .	81
6.4.10	Output Selection Lists . . . . .	81
6.4.11	Steady Aerodynamics. . . . .	81
6.4.12	Structural Element Connection. . . . .	82
6.4.13	Structural Element Properties. . . . .	82
6.4.14	Unsteady Aerodynamics. . . . .	83
6.4.15	Discipline Dependent Problem Control . . . . .	83
6.5	Summary of ASTROS/NASTRAN Bulk Data Entry Differences. . . . .	83
VII	ASTROS OUTPUT FEATURES. . . . .	87
7.1	System Controlled Output . . . . .	88
7.1.1	Default Output Printed by Modules . . . . .	88
7.1.2	Error Message Output. . . . .	90
7.2	Quantities Selected Through Solution Control Options . . . . .	91
7.2.1	Element Response Quantities . . . . .	93
7.2.2	Nodal Response Quantities . . . . .	107
7.2.3	Design Variable and Design Constraints. . . . .	109
7.2.4	Flutter/Normal Modes Response Quantities. . . . .	111
7.2.5	Aeroelastic Trim Quantities . . . . .	112
7.3	Executive Sequence Selectable Quantities . . . . .	115
7.3.1	Bulk Data Echo Option . . . . .	115
7.3.2	Intermediate Steady Aerodynamic Matrix Output . . . . .	116
7.3.3	Intermediate Unsteady Aerodynamic Matrix Output . . . . .	117
7.3.4	Flutter Root Iteration Output . . . . .	118
7.3.5	Stress Constraint Computation Output. . . . .	118
7.3.6	Intermediate optimization Output. . . . .	119

## TABLE OF CONTENTS (Continued)

<u>SECTION</u>	<u>PAGE</u>
7.4 Executive Sequence Output Utilities. . . . .	119
7.4.1 Structural Set Definition Utility, USETPRT. . . . .	120
7.4.2 Special Matrix Print Utility, UTGPRT. . . . .	120
7.4.3 General Matrix Print Utility, UTMPRT. . . . .	121
7.4.4 General Relation Print Utility, UTRPRT. . . . .	121
7.4.5 General Unstructured Print Utility, UTUPRT. . . . .	122
7.5 ASTROS Output Limitations. . . . .	122
REFERENCES. . . . .	125

<u>APPENDIX</u>	<u>PAGE</u>
A ASSIGN DATA BASE DESCRIPTIONS . . . . .	127
A.1 VMS Implementation . . . . .	128
A.2 IBM 370 Implementation . . . . .	130
A.3 Perkin Elmer Implementation. . . . .	131
B THE MAPOL PROGRAMMER'S MANUAL . . . . .	135
C ANNOTATED STANDARD EXECUTIVE SEQUENCE . . . . .	179
C.1 Variable Declarations. . . . .	180
C.2 Preface Modules. . . . .	180
C.3 Optimization Phase . . . . .	181
C.4 Design Convergence Loop. . . . .	182
C.5 Optimization Phase Boundary Condition Loop . . . . .	182
C.6 Active Constraint Selection. . . . .	190
C.7 Sensitivity Evaluation for Math Programming. . . . .	190
C.8 Final Analysis Phase . . . . .	196
C.9 Analysis Phase Boundary Condition Loop . . . . .	196
D SOLUTION CONTROL COMMANDS . . . . .	231
E BULK DATA ENTRIES . . . . .	253

# LIST OF ILLUSTRATIONS

<u>FIGURE</u>		<u>PAGE</u>
1	The General ASTROS Input Data Stream. . . . .	6
2	Function of the ASSIGN DATABASE Entry . . . . .	9
3	Structure of the Standard MAPOL Algorithm . . . . .	20
4	Multidisciplinary Optimization Flowchart. . . . .	21
5	Bulk Data Entry Format. . . . .	76
6	BAR Element Coordinate System . . . . .	96
7	BAR Element Force Sign Convention . . . . .	96
8	IHEX1 Element Stress Computation Points . . . . .	99
9	IHEX2 Element Stress Computation Points . . . . .	99
10	IHEX3 Element Stress Computation Points . . . . .	101
11	ROD Element Coordinate System . . . . .	102
12	QDMEM1 Element Coordinate System. . . . .	104
13	QUAD4 Element Coordinate System . . . . .	104
14	SHEAR Element Coordinate System . . . . .	106
15	TRMEM Element Coordinate System . . . . .	107
C-1	Standard MAPOL Sequence . . . . .	204

## LIST OF TABLES

<u>TABLE</u>		<u>PAGE</u>
1	Keywords Recognized by the Input Stream Interpreter . . . . .	7
2	ASTROS Debug Packet Commands . . . . .	12
3	MAPOL Edit Commands . . . . .	18
4	Real Parameters in the Standard Sequence . . . . .	23
5	Integer Parameters in the Standard Sequence . . . . .	24
6	Discipline Options Matrix . . . . .	62
7	Response Quantity Print Options Matrix . . . . .	70
8	Aerodynamic and Structural Elements in ASTROS . . . . .	94

## SECTION I

### INTRODUCTION

This User's Manual is one of four manuals documenting the ASTROS (Automated STRuctural Optimization System). The other three are the Applications Manual, the Programmer's Manual and the Theoretical Manual. The Applications and Theoretical Manuals indicate the range of capabilities of the ASTROS system while the Programmer's Manual is provided to give details of the internal workings of the engineering modules. The User's Manual provides a complete description of the user interface to the ASTROS system in order to facilitate the preparation of input data. It introduces the features of the ASTROS system that enable the user to direct this software system and documents the mechanisms by which the user can communicate with the system. It is assumed that the reader is familiar, from a study of the Theoretical Manual, with the engineering capabilities of the ASTROS system and is using this manual to define the form of the particular input that directs the system to perform a desired function. This manual is intended to provide the user with a convenient reference for all forms of input to the system and is therefore organized along the same lines as the input data stream. The discussion of each topic is brief and as generic as possible with the detailed documentation isolated in the appendices. Information on ASTROS output formats is also provided.

This manual is directed toward the engineer/designer/analyst who is using ASTROS to perform engineering design or analysis. While ASTROS is perfectly capable of performing many tasks not explicitly supported in the standard execution, the user must know the engineering software in considerable detail to direct the procedure to perform these alternative functions. The mechanisms by which these more advanced features are invoked are included in this manual but no attempt is made to provide sufficient information to the user to generate new analysis features or to grossly modify the existing capabilities of the system. These more advanced topics are treated in the Programmer's and Application Manuals which document the individual modules in the system and their interactions. Rudimentary (and typical) modifications to the execution sequence and changes to execution parameters are discussed in detail in this manual.

Machine and installation dependent aspects of ASTROS are also contained in the Application and Programmer's Manuals rather than in the User's Manual. Only those machine dependency issues that are logically related to the preparation of the input are discussed in the User's Manual. Machine dependencies in the input are limited to the naming conventions for the run time data base file(s) and the parameters that can be used on the ASSIGN DATABASE entry. Other machine dependencies are handled as part of the installation of the system on each particular host machine. These issues are documented in the Programmer's Manual since they are relevant only to the "system manager," not to the "user."

It will be apparent to many readers that the NASTRAN structural analysis system was used as a guide in the design of the ASTROS procedure. Both NASTRAN and ASTROS comprise large scale, finite element structural analysis in executive driven software systems. Therefore, many of the input and output features are similar. NASTRAN has become an industry standard in finite element structural analysis with many pre- and post-processors developed around NASTRAN data. In order to facilitate acceptance and use of the ASTROS procedure, NASTRAN compatibility was considered desirable in many areas. As a result, many aspects of the ASTROS input are similar in form or purpose to those in NASTRAN and, in many other cases, the same nomenclature has been adopted. Therefore, in some instances in this document, ASTROS input will be compared and contrasted to NASTRAN input in order to present a concise picture of the ASTROS input and to assist the reader familiar with NASTRAN in making the connection to the equivalent item in ASTROS. Although familiarity with NASTRAN is not a prerequisite to understanding the ASTROS documentation, sufficient numbers of potential ASTROS users are expected to be familiar with the NASTRAN system to justify the sometimes casual reference to NASTRAN features.

Section II contains a description of the ASTROS input file with the remaining Sections III through VI organized to parallel the input file structure. Within each section, the function of the particular input packet is presented along with illustrations of how the data are prepared. Each packet is described in a generic fashion so as to indicate how the sophisticated user can make use of the more advanced features of the system without cluttering



the discussion with details of the input structures. The detailed documentation of the separate input structures of the data packet are in an accompanying appendix. This form of user's documentation enables this manual to be useful as a guide to the beginning user as well as a reference for the experienced user.

In Section VII, following the input stream descriptions, the output features of the ASTROS system are documented. While these features are selected through directives in the input data stream, they are sufficiently numerous and complex to justify a separate section devoted solely to output requests. The output capabilities of the system are described in very general terms while the output requests available for each analysis discipline and optimization feature are documented in detail. Most output is selected through Solution Control directives that are documented in Section V, but some are selected through modifications to the executive (MAPOL) sequence. Section VII documents all of the output utilities available to the user through MAPOL directives and gives several examples of modifying the MAPOL sequence to obtain additional output. Other features are described in the MAPOL Programmer's Manual.

Many examples of user input are used throughout this document and the appendices. In order to ease the burden of interpretation, the following conventions will be used in the examples unless otherwise noted.

MAPOL NOGO	Capital letters indicate that the phrase must appear exactly as shown
MAPOL params	Lower case symbols act as generic place holders indicating that an option or options can or must be included
MAPOL { <u>GO</u> } { NOGO }	Symbol(s) enclosed in braces ( ) are optional If more than one symbol is available they be stacked in vector notation with any defaults denoted by underlines.
INCLUDE <filename>	A required symbol is enclosed in angle brackets. If the angle brackets surround an option list, at least one of the available options must be selected.
BEGIN ^ BULK	The caret (^) is used to signify a required blank space.

In some cases, the appendices represent standalone documents with their own conventions that may differ from those in the main body.

## SECTION II

### THE INPUT DATA STREAM

The ASTROS user directs the system through an input data stream composed of a command to attach the ASTROS run time data base followed by multiple data packets. Each packet contains a set of related data providing the information needed to execute the ASTROS procedure. The packets begin with a keyword indicating the nature of the data within the packet and terminate with an ending keyword or with the start of the next data packet. All the packets in the input data stream are optional, although the order in which they must appear is fixed. The purpose of this section is to document the structure of the input data stream. Detailed documentation of the data within each data packet is then presented in separate sections.

Figure 1 shows the general form of the input data stream. The first non-blank record of the input file must be the ASSIGN DATABASE entry. This command enables the user to attach the run time data base file(s) that are used during the execution of the ASTROS procedure. There are four optional data packets following the ASSIGN DATABASE entry which, if they are present, must appear in the order shown. The first is the DEBUG packet which contains "debug" commands to control or select specific actions within the executive and data base management systems. The second packet is the MAPOL packet containing the executive system control directives consisting of either a standalone MAPOL program or EDIT commands to modify the standard MAPOL program. If the MAPOL packet is absent, the unmodified standard MAPOL sequence directs the execution. The Solution Control commands appear in the third optional packet denoted by the keyword SOLUTION. These commands select the engineering data to be used in each subcase from the set of data provided in the fourth and final data packet: the BULK DATA packet. The BULK DATA packet contains the engineering data describing the finite element structural model, the aerodynamic model(s), and the design model, as well as all the data needed to perform the specific analysis and/or optimization tasks. The MAPOL, SOLUTION and BULK DATA packets are analogous to the NASTRAN executive control, case control and bulk data decks, respectively.

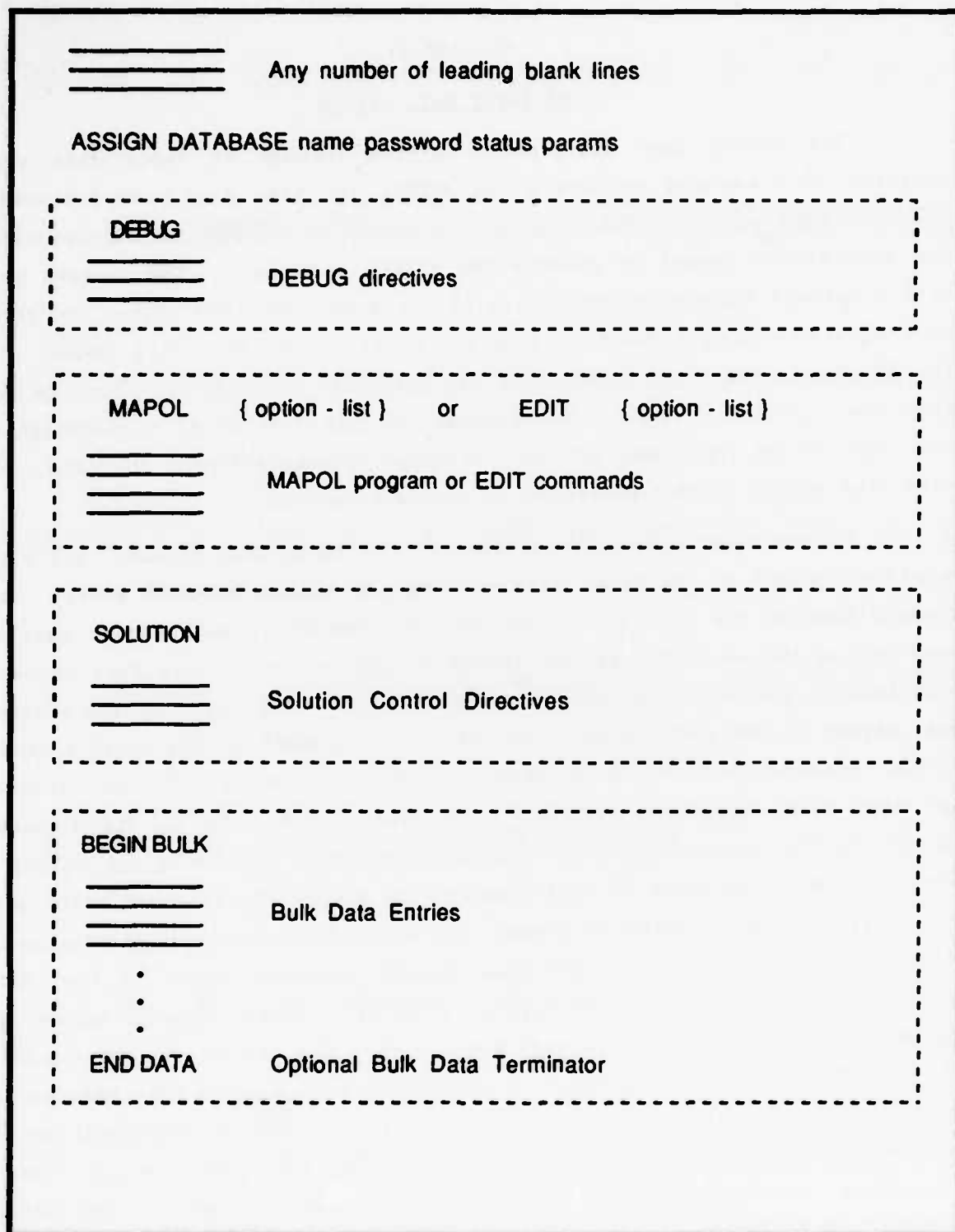


Figure 1. Structure of the ASTROS Input Data Stream

In interpreting the input data stream, ASTROS recognizes the keywords shown in Table 1. These keywords must begin in the first column and have the structure shown. In some cases the keyword is also a command line that makes up part of the data packet which it initiates. In these cases, the command parameters are documented in the User's Manual section discussing the details of the associated data packet. For example, the MAPOL keyword is part of a command that directs the MAPOL compiler to take certain actions. The detailed discussion of the MAPOL command is therefore contained in the MAPOL Programmer's Manual (Appendix B of this report) and in Section IV.

TABLE 1. KEYWORDS RECOGNIZED BY THE INPUT STREAM INTERPRETER

KEYWORD	DEFINITION
ASSIGN DATABASE	Identifies the run-time data base files
DEBUG	Initiates the DEBUG Packet
MAPOL	Command line which initiates a MAPOL packet containing a complete, user defined MAPOL program.
EDIT	Command line which initiates a MAPOL packet containing edit commands modifying the standard executive sequence.
SOLUTION	Initiates the SOLUTION CONTROL packet.
BEGIN (BULK)	Initiates the BULK DATA packet.
ENDDATA	Optionally terminates the BULK DATA packet.
INCLUDE	Directs ASTROS to include input data from additional files.

Two keyword commands are related only to the input data stream and not to the data within a packet. These are the ASSIGN DATABASE and the INCLUDE commands. Each of these is discussed in detail in the following subsections.

#### 2.1 THE ASSIGN DATABASE ENTRY

The first line of the ASTROS input data stream must be the ASSIGN DATABASE entry. This entry identifies the run time data base files to be used in the current ASTROS execution and specifies certain parameters associated with the files. The format of this entry is:

ASSIGN DATABASE <dbname> <password> <status> (params)

where,

dbname	is a name identifying the run time data base files (maximum of 8 characters)
password	is a user assigned password for the data base files (maximum of 8 characters)
status	is the status of the data base files. Must be either OLD, NEW or TEMP
params	are optional (installation dependent) parameters e.g., DBLKSIZ = n, IBLKSIZ = n, etc.

The entries on the ASSIGN DATABASE command are NOT keyword controlled and so must be input in the order shown. They can be separated by any number of blank characters or commas but must reside on one physical record of the input data stream. Note that two commas do not imply that a parameter is missing; instead, the second comma will be ignored. Two equivalent examples are shown below:

ASSIGN DATABASE, ASTSC, KIMBERLY, OLD

ASSIGN DATABASE ASTSC KIMBERLY OLD

Figure 2 is provided to illustrate the function of the ASSIGN DATABASE entry. In the case shown, the installation dependent parameter VOL has been implemented which allows selection of the physical device on which the requested file(s) DB1 reside.

The optional "params" on the ASSIGN DATABASE entry may or may not be keyword controlled and are installation dependent. They provide a mechanism for the user to direct machine or installation dependent file operations to be performed by the ASTROS procedure. At each site, the installation of the code involves a definition of these parameters and the form they must take. The ASTROS procedure is currently functional on five host systems: IBM 370 series, Perkin Elmer 30XX series, VAX/VMS and MicroVMS, VAX/Ultrix and CRAY. Appendix A documents the installation dependent ASSIGN DATABASE parameters for most of these systems as defined by the ASTROS system installer at each site. The Programmer's Manual contains the detailed description of how these and other machine dependent parameters are defined.

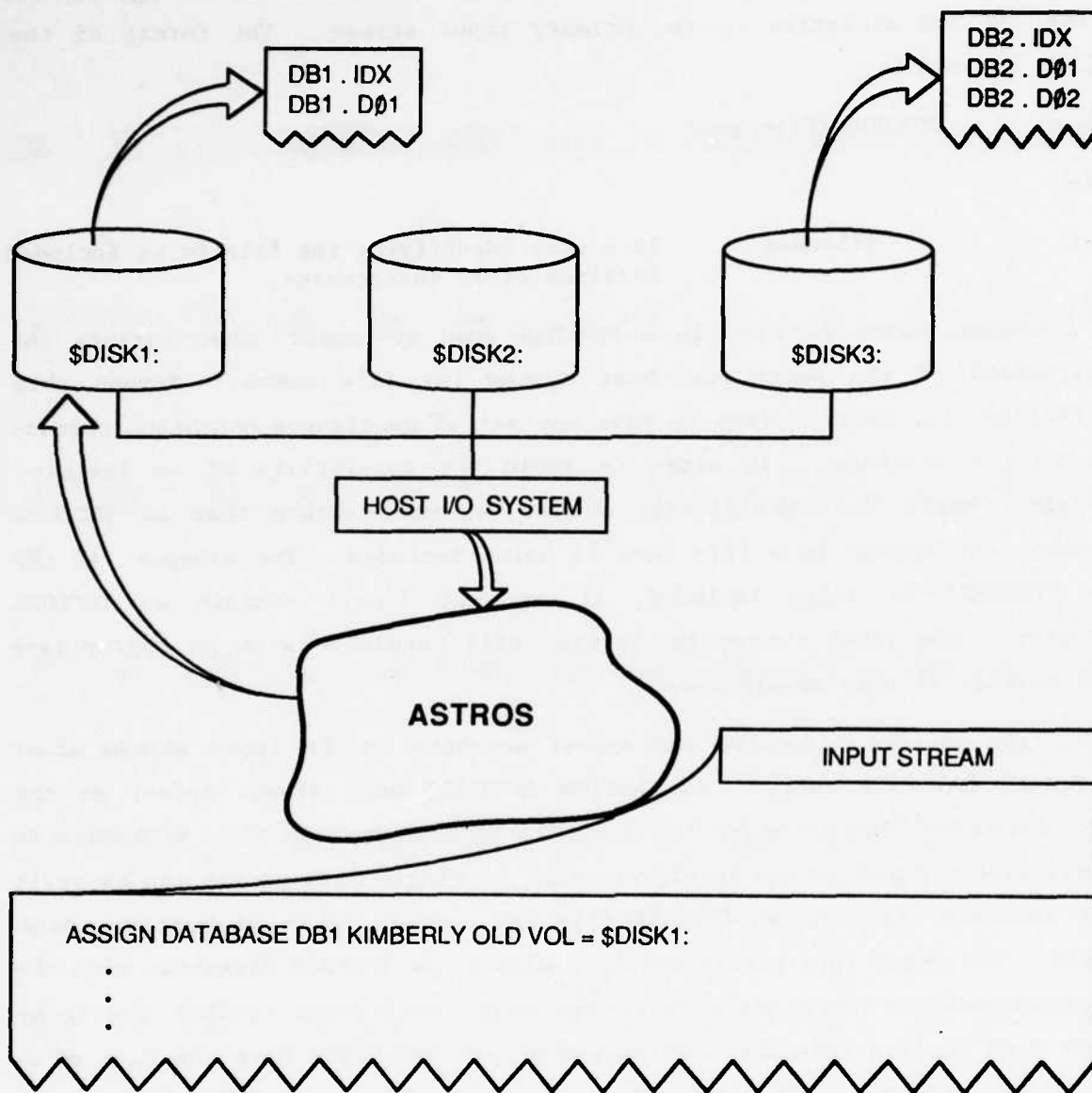


Figure 2. Function of the ASSIGN DATABASE Entry

## 2.2 THE INCLUDE DIRECTIVE

The input data stream typically resides in a single file, but the user can direct the input stream interpreter to include other files through the use of the INCLUDE directive in the primary input stream. The format of the INCLUDE command is:

```
INCLUDE <filename>
```

where,

filename	is a name identifying the file to be included (maximum of 72 characters).
----------	--

The filename, which is used in a FORTRAN open statement, must satisfy the requirements of the particular host system for file names. Beyond this restriction, the user is free to have any set of contiguous non-blank characters in the filename. In order to avoid the possibility of an infinite recursion, there is a restriction on the include feature that no INCLUDE statement can appear in a file that is being included. For example, if the file "TENBAR" is being included, it may not itself contain an INCLUDE directive. The input stream interpreter will terminate with an appropriate error message if this should occur.

The INCLUDE directive can appear anywhere in the input stream after the ASSIGN DATABASE entry. The ASSIGN DATABASE must always appear as the first non-blank line in order to allow the use of the run time data base in the subsequent input stream interpretation. A single data packet can be split among included files or an INCLUDE file may contain parts of multiple data packets. The input interpreter merely replaces the INCLUDE directive with the data contained in the named file so the only requirement is that the input stream that results from the combination of all INCLUDE's have the form of a normal input stream. The INCLUDE feature can be very useful in certain circumstances. For example, a special user developed MAPOL sequence can be stored and externally maintained from the files containing the engineering data for particular runs; or, conversely, the bulk data representing a large model can be included into the file containing the solution control directives.



### SECTION III

#### THE DEBUG PACKET

The debug packet represents a legitimization of a development tool and is intended to be used primarily by those responsible for maintaining the software. The debug packet provides the system programmer with the means to invoke or control certain executive and data base management system functions that are helpful in tracking the ASTROS execution and/or testing the executive and data base management system software. However, because some of the debug options can be useful to the general user, the debug packet is fully documented in the User's Manual rather than in the Programmer's Manual. This section documents each of the debug options and indicates how the option can be useful in debugging the ASTROS procedure. Emphasis is placed on those debug options that are of interest to the general user.

The debug packet is initiated by the keyword DEBUG, which must appear alone on the line of the input stream that follows the ASSIGN DATABASE entry and that precedes any other data packet. Following the initiator, any number of debug lines can be included in the data stream. Each debug command line can be composed of a number of debug commands, appearing in any order, separated by blanks or commas. The DEBUG packet is terminated when a new data packet initiator, or the end of the input stream, is encountered. Most debug commands consist of single keywords which toggle flags activating the debug functions. The appearance of these debug keywords is all that is required to activate the option. Other debug commands select that a flag take on a particular value. These commands have the form:

`<command> = <value>`

There can be any number of blanks between the end of the command keyword, the value and the equal sign, but neither the command nor the value can contain imbedded blanks. Any errors in the debug packet input will result in warnings but will not terminate the execution and the erroneous command will be ignored.

Table 2 shows the list of keywords that can be included in the DEBUG packet. The debug commands are grouped into executive system and data base

TABLE 2. ASTROS DEBUG PACKET COMMANDS

EXECUTIVE SYSTEM DEBUGS

KEYWORD	DESCRIPTION
MSTACK	MAPOL stack output.
MEXEC	Execution debugs for the MAPOL code.
MOBJ	Object code debugs for the MAPOL compilation.
MTRACE	MAPOL instruction trace output.
MATRIX	Large matrix utility trace.

DATA BASE/MEMORY MANAGEMENT DEBUGS

KEYWORD	DESCRIPTION
TRACE	Data base call tracing.
EVENT	Data base event tracing.
BUFFER	Data base buffer dump.
IOSTAT-<parm>	Data base input/output tracing. <parm> is either FULL for a complete trace and statistical summary or SUM to limit the output to the summary of I/O statistics.
ENTITY-<name>	Specifies the name of a data base entity which limits the TRACE, EVENT, BUFFER, and IOSTAT-FULL options to be active only when the named entity is open.
CALLSTAT	Compiles a summary of data base calling statistics for output on termination of the execution.
NOCOREDİR	Stores the data base directories on the data base files rather than in memory.
MEMORY	Memory management call tracing and checksum.

management system debugs. Each of these groups is described in greater detail in the following subsections.

### 3.1 EXECUTIVE SYSTEM DEBUG COMMANDS

The first four executive system keywords in Table 2 are intended to assist the system programmer in following the actions of the MAPOL compiler and execution monitor. As such, they are of limited value to the general user. The MATRIX option, however, can be useful in tracking the execution of the MAPOL program. It echoes the matrix utility calls for all matrix operations that are in the MAPOL sequence. For example, if the MAPOL program includes the expression:

$$[A] := \text{TRANS}([B]) * [C] + [D];$$

the MATRIX trace echoes the resultant call to the MPYAD large matrix utility with the arguments shown in detail. This trace can be very useful in determining which particular MAPOL instruction is being executed when a problem occurs. Large MAPOL programs with many loops and a large number of matrix expressions can be debugged quite simply using the MATRIX trace. All MAPOL statements that result in calls to any of the large matrix utilities (e.g., PARTN, MERGE, MPYAD, MXADD, etc.) are echoed.

### 3.2 DATA BASE AND MEMORY MANAGER DEBUG COMMANDS

The data base management system has a number of debug options which can be divided into three categories: trace options, control options and memory manager options. The first group in Table 2 contains a number of tracing options: TRACE, EVENT, BUFFER, and IOSTAT. The IOSTAT keyword selects either a FULL tracing or a SUMMARY. The first three of these options and the IOSTAT-FULL option are further controlled by the ENTITY option which completes the first group of keywords. Note: the tracing keywords generate an overwhelming amount of data which are often of limited use unless the user is familiar with the internal structure of the data base files. The ENTITY keyword limits the activation of the tracing options to those times when the named data base matrix, relation or unstructured entity is open. If no entity specification is made, the traces are active for all data base operations. In addition to their role in debugging the data base software, the trace options provide a useful means of debugging the interface between a user written module and the data base.

The data base control options CALLSTAT and NOCOREDIR provide user control over two internal data base functions. The CALLSTAT option compiles a summary of the number of calls made to each data base subroutine. This summary, in combination with the IOSTAT option, provides statistics on the number of data base operations in the execution. The NOCOREDIR option is made available for machines with limited core memory resources. If NOCOREDIR is selected, the data base manager stores the data base directories on the data base files rather than in core. This can substantially reduce the data base memory requirements at the cost of increasing the number of input/output operations.

The last "group" of data base debug options consists of the MEMORY command. This option causes an echo of all the memory management calls made in the modules. The user can then track the ASTROS execution into the engineering modules themselves. In addition to the echo, the MEMORY option invokes a "checksum" operation which checks for the integrity of the memory block headers on every memory manager operation. If the checksum fails, a message is written to the effect that a block header has been overwritten. This option is very effective in uncovering errors in engineering modules that make use of dynamic memory allocation.

## SECTION IV

### THE EXECUTIVE SYSTEM (MAPOL PACKET)

The ASTROS system is an executive controlled software procedure. One of the functions of the ASTROS executive system, described in detail in Reference 1, is to determine the sequence in which the modules of the procedure are invoked. For ASTROS, the Matrix Abstraction Problem Oriented Language (MAPOL) has been developed to perform this executive system task. The MAPOL language has its conceptual roots in the Direct Matrix Abstraction Program (DMAP) capability developed for the NASTRAN structural analysis system (Reference 2). The DMAP "language," used to create NASTRAN's solution algorithms is very crude; however, it has been a major factor in extending the life cycle of the software. It provided a simple method of installing new code and functional capabilities into the system and afforded the user an opportunity to interact with the software. MAPOL provides the same advantages to the ASTROS system and represents a considerable advance over DMAP in that MAPOL is a structured, procedural language that directly supports high order matrix operations, manipulation of data base entities and complex data types. Moreover, the syntax of the language looks much like that of any scientific programming language and so is easily learned by anyone who knows FORTRAN or PASCAL.

From the user's point of view, ASTROS is directed by a sequence of control statements "coded" in the MAPOL language just as a NASTRAN rigid format is coded in the DMAP "language." The executive system within ASTROS compiles the MAPOL program and executes the resultant "ASTROS machine code" which directs the execution of the ASTROS procedure. (Note that ASTROS is NOT written in MAPOL, only the executive control algorithm is written in the MAPOL language. In fact, ANSI standard FORTRAN was used to write the compiler for MAPOL and all the engineering software of the ASTROS system.) This control language allows the user to manipulate the software system in many ways to tailor the available capabilities to perform particular tasks. At a higher level of sophistication, the user may add modules to the system or replace modules that already exist. Obviously, some of these features require a knowledge of the ASTROS system that is beyond the scope of the User's Manual. Those features that require detailed information are more fully discussed in

the Programmer's Manual but their existence is emphasized here in order to introduce the user to the flexibility that the executive system provides.

The remainder of this section serves two functions. First, it presents the mechanics of the MAPOL packet and second, it presents the standard MAPOL sequence that has been developed to direct the optimization and analysis tasks for which ASTROS has been designed. The potential of the executive system to tailor the ASTROS procedure will be explored in this discussion of the standard sequence. In support of this section, Appendix B is a standalone document that presents the MAPOL language, its syntax and features while Appendix C contains an annotated copy of the ASTROS standard executive sequence. It cannot be overemphasized that, while the capabilities implemented in the ASTROS software are significant, the true power embodied in the ASTROS system is its immense flexibility, largely provided by the executive system and its MAPOL language.

The MAPOL packet is initiated either by the keyword MAPOL or the keyword EDIT and is terminated upon encountering the SOLUTION CONTROL packet, the BULK DATA packet or the end of the input stream. In addition, each of the initiator keyword commands act as directives to the MAPOL compiler to take specific actions. The exact form of the MAPOL and EDIT commands is:

---

$$\text{MAPOL } \left\{ \begin{array}{c} \text{GO} \\ \text{NOGO} \end{array} \right\} \left\{ \begin{array}{c} \text{LIST} \\ \text{NOLIST} \end{array} \right\} \quad \text{or} \quad \text{EDIT } \left\{ \begin{array}{c} \text{GO} \\ \text{NOGO} \end{array} \right\} \left\{ \begin{array}{c} \text{LIST} \\ \text{NOLIST} \end{array} \right\}$$

where:

GO/NOGO	selects whether the MAPOL program is to be executed after compilation.
LIST/NOLIST	selects whether the MAPOL source code is to be written to the output file.

---

The MAPOL command is followed by a MAPOL program which can be any syntactically complete set of MAPOL statements as described in the MAPOL Programmer's Manual (Appendix B). The EDIT command indicates that the MAPOL packet will consist of edit commands that insert, delete or replace lines of the standard executive sequence.

#### 4.1 THE MAPOL PROGRAM

If the MAPOL packet begins with the MAPOL command line, the compiler assumes that the remaining statements in the packet constitute a complete MAPOL program. That program can be any set of MAPOL statements that satisfy the rules of the language as presented in Appendix B. The program can call any of a number of intrinsic functions (including most of the common FORTRAN intrinsic functions) and any of the "engineering" utilities and modules that have been defined to the compiler. The user can access these modules in any desired order, subject only to limits imposed by the engineering modules themselves. In addition, the user can write special purpose modules and define them to the compiler through the SYSTEM GENERATION (SYSGEN) program discussed in the Programmer's Manual. Thus, a wide range of tasks can be performed using the ASTROS system in combination with a user's MAPOL program.

The MAPOL language can be read and written easily by anyone familiar with a scientific programming language. This feature opens the advantages of the executive system to the average user without requiring specialized knowledge in computer science or requiring effort to learn a radically different programming language. The user will often find the simplicity and power of the MAPOL language enables many tasks to be performed using the ASTROS system that are not explicitly supported in the standard executive sequence.

#### 4.2 MAPOL EDIT COMMANDS

If the MAPOL packet begins with the EDIT command line, the compiler assumes that the remainder of the packet (if any) is composed of MAPOL edit commands and new MAPOL statements that modify the standard executive sequence. The set of edit commands is given in Table 3 (and in Appendix B). They allow the user to insert, delete and replace lines of the standard MAPOL sequence. All the edit commands reference a line number or range of line numbers. The line numbers are those in a compiled listing of the standard MAPOL sequence which is written as part of the system generation task. When editing the standard sequence, the user is cautioned to obtain the most recent listing either from the SYSGEN output or by executing ASTROS with an input stream containing only an ASSIGN DATABASE entry and the one line MAPOL packet:

EDIT LIST NOGO

This input stream will result in an output file containing the current listing of the standard executive sequence.

TABLE 3. MAPOL EDIT COMMANDS

STATEMENT	FUNCTION
EDIT	Modify the standard solution.
DELETE a[,b]	Remove lines a through b inclusive.
REPLACE a[,b]	Remove lines a through b inclusive and replace with following lines.
INSERT a	Insert the lines following the command after line a.

#### 4.3 THE STANDARD EXECUTIVE SEQUENCE

As previously mentioned, the MAPOL language has its conceptual roots in the DMAP "language." To allow the user of NASTRAN to perform certain predefined analyses, a set of "rigid formats" or DMAP algorithms were written, alleviating the user of the need to learn the details of the control language. Each rigid format allowed the user to perform analyses in a different engineering discipline; for example, static structural analyses, normal modes analyses, or transient analyses. In a similar manner, a standard executive sequence or MAPOL algorithm has been developed for the ASTROS system which supports all the engineering disciplines and optimization features of the procedure. Unlike the multiple DMAP rigid formats, however, there is a single MAPOL sequence that supports all the available engineering disciplines as well as optimization. This fundamental difference is necessary to permit multidisciplinary optimization.

One consequence of having a single multidisciplinary algorithm is that the standard sequence appears to be very complicated. The purpose of this subsection is to present the internal structure and flow of the standard MAPOL sequence, thereby providing the user with sufficient information to tailor the standard sequence to suit individual needs. The discussion in this section will be general to provide the necessary overview and to introduce the



concepts embodied in the standard sequence. Modifications to the standard sequence are presented primarily in terms of capabilities, but the presentation is supported by examples that represent both simple and more complex modifications. Appendix C supplements this discussion with a detailed line-by-line presentation of the standard executive sequence. The reader is also referred to the Programmer's Manual for information on the addition of modules to the ASTROS engineering library.

#### 4.4 STANDARD EXECUTIVE SEQUENCE STRUCTURE

The standard MAPOL sequence consists of two major components: the variable declarations and the solution algorithm. The solution algorithm can be further divided into preface modules, the optimization segment and the final analysis segment. The declaration segment declares all variables used in the MAPOL sequence. This includes all integer and real scalar variables as well as high order variables: relations, matrices and unstructured data base entities. Within the solution algorithm, the preface modules comprise a group of engineering modules exercised prior to the boundary condition loops to perform a number of system initialization tasks; e.g., loads generation and the computation of invariant aerodynamic matrices. The separate optimization and analysis segments consist of a loop on the number of (optimization or analysis) boundary conditions in the current execution. In the optimization segment, a second boundary condition loop is performed to obtain the sensitivities of active boundary condition dependent constraints in preparation for the optimization task.

Figures 3 and 4 juxtapose the standard algorithm structure and a flow chart showing how multidisciplinary optimization is performed in ASTROS. It is readily apparent that the structure of the standard MAPOL sequence has been determined by the requirement to perform multidisciplinary optimization. Each of the segments of the standard sequence are discussed in greater detail in the following subsections.

##### 4.4.1 MAPOL Declarations

MAPOL is a strongly typed language that requires all variables used in a program unit (either the main program or a procedure) to be declared. This applies to both simple variables like real and integer scalar or array variables and to high order variables (like MATRIX) that refer to data base entities. The first several hundred lines of the standard sequence consist solely

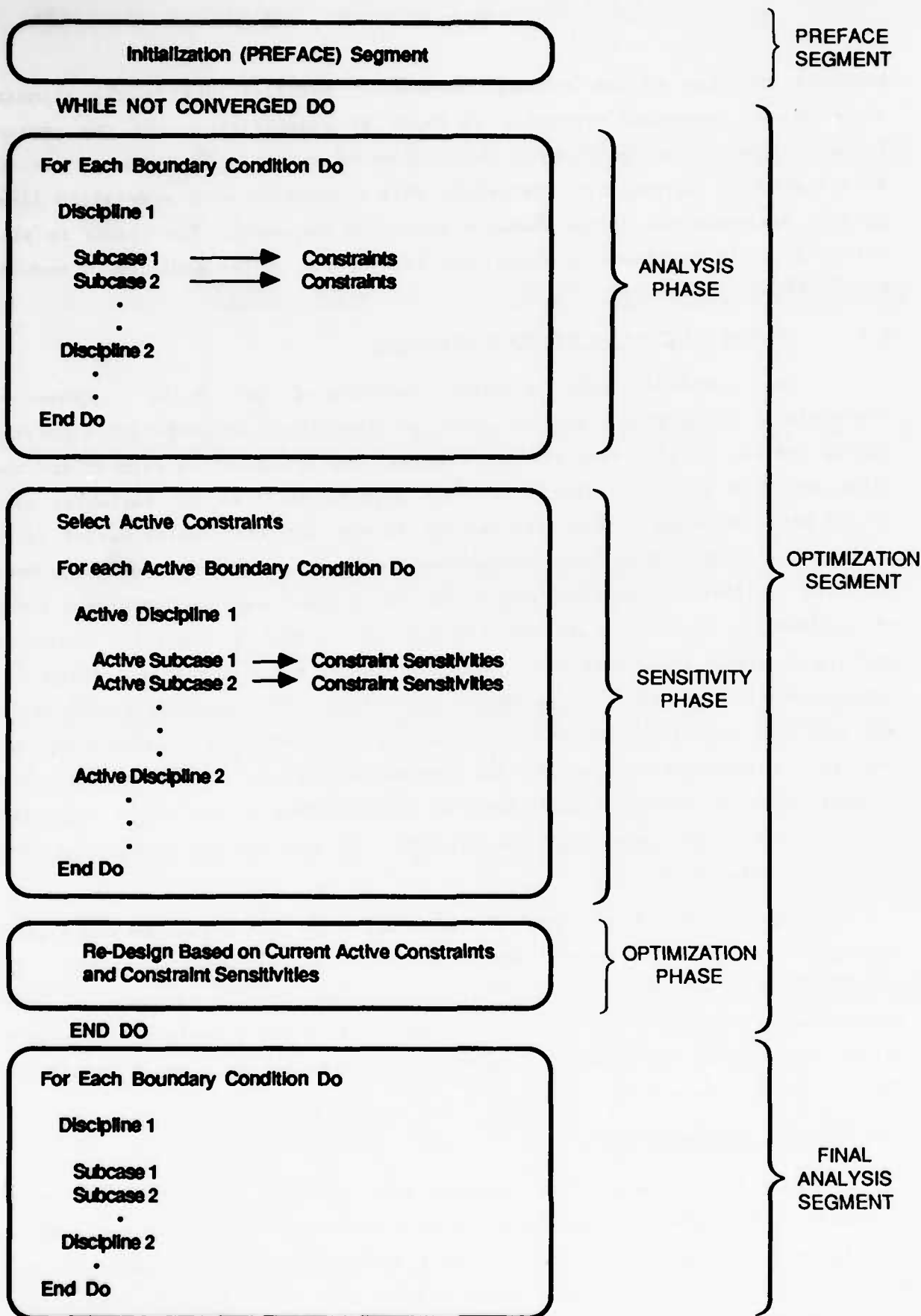


Figure 3. Structure of the Standard MAPOL Sequence.

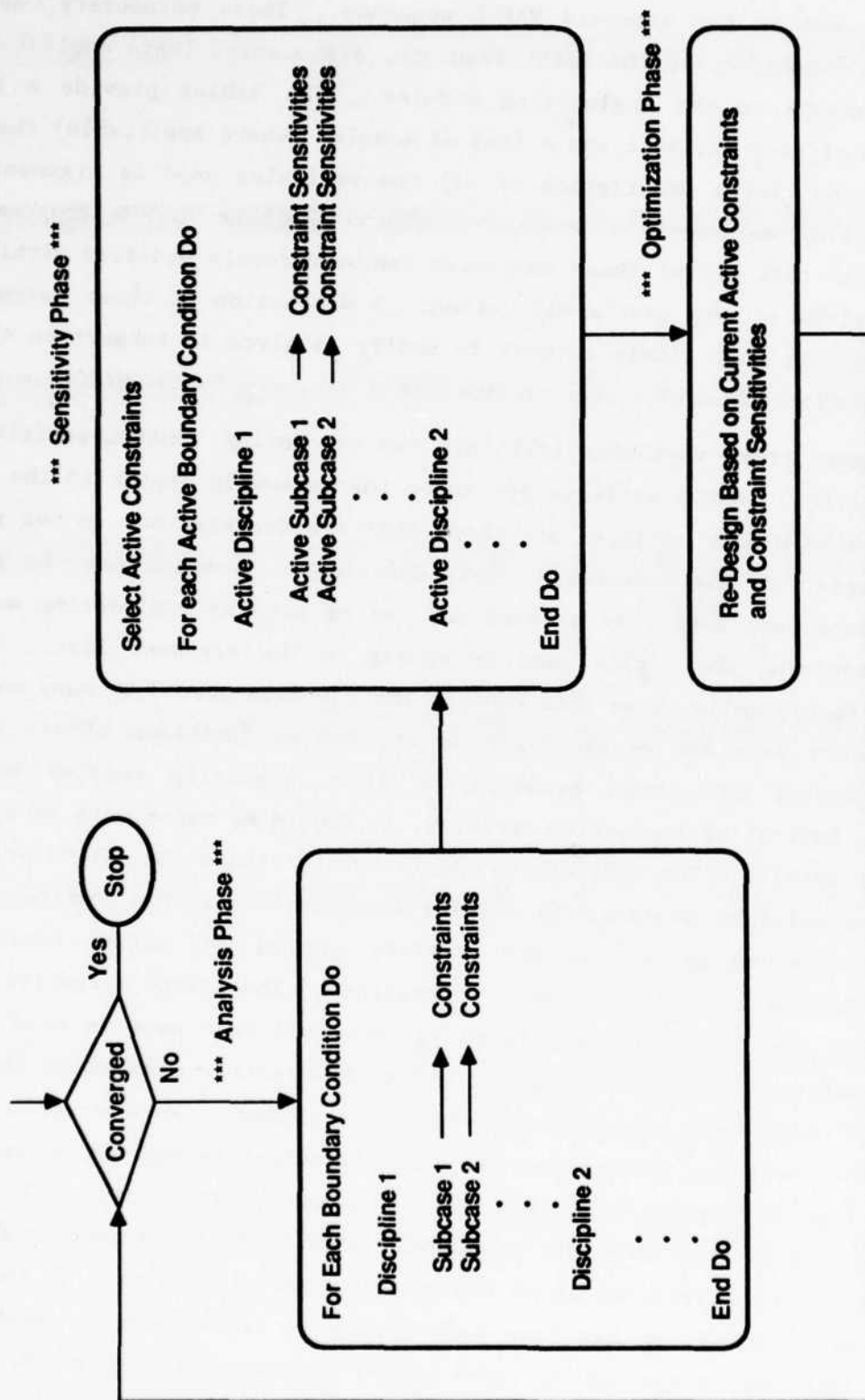


Figure 4. Multidisciplinary Optimization Flow Chart.

of these variable declarations. Tables 4 and 5 show the list of scalar parameters used in the standard MAPOL sequence. These parameters, set in engineering modules or in the MAPOL sequence, are used as logic control flags and/or arguments to the engineering modules. The tables provide a brief description of each variable and a list of modules (where applicable) that use the parameter. For a description of all the variables used as arguments of the engineering modules, the reader is referred to the ASTROS Programmer's Manual. Note that all of these variables can be directly modified within the MAPOL algorithm at the user's discretion. A discussion of those parameters that the user is most likely to want to modify is given in Subsection 4.4.3, but the experienced user is free to change any variable in the MAPOL sequence.

Higher order variables fall into two categories: MAPOL entities and HIDDEN entities. MAPOL entities are those that actually appear in the MAPOL sequence while HIDDEN entities are those that are declared but do not subsequently appear in the sequence. Their declaration ensures that the corresponding data base entity is created and can be used by engineering modules without requiring the entity name to appear in the argument list. HIDDEN entities are typically those that contain the raw data needed by many modules; e.g. geometry data and connectivity data. The declarations of the higher order variables have been arranged to place logically related entities together. Several of the matrix entities, it should be noted, are subscripted (e.g. [KLLINV(8)]). The subscripted matrix entity allows the ASTROS procedure to perform multiple analyses in several boundary conditions and retain the information needed to compute the sensitivities of the active constraints retained from each of these boundary conditions. The ASTROS executive system generates a name for each subscripted variable and that name is used by all the engineering modules receiving the subscripted entity name as an argument. The actual data base entity name need not be known. This does, however, impose the following restriction: a subscripted entity may not be used as a HIDDEN entity in any engineering module; it must appear in the calling list for the module because only the executive system knows the actual name of the data base entity corresponding to the current subscript value. In the standard sequence, provision has been made for up to eight boundary conditions, but the user can change the declared number of subscripts to match the required number of boundary conditions.

TABLE 4. REAL SCALAR PARAMETERS IN THE STANDARD SEQUENCE

NAME	MODULES	DESCRIPTION
ALPHA	FSD	Exponential move limit for fully stressed design. (Def = 0.90)
BQDP	BLASTFIT, others	Dynamic pressure value used in the current nuclear blast subcase. Output from BLASTFIT, used subsequently in MAPOL expressions.
CNVRGLIM	DESIGN, FSD	Convergence test limit specifying the maximum percent objective change for the approximate problem to be considered converged. (Def=0.50)
CTL	ACTCON, DESIGN, FSD	Criteria for denoting a constraint to be active in determining convergence in ACTCON. If value > CTL, the constraint is active. Set in DESIGN or in FSD.
CTLMIN	ACTCON, DESIGN, FSD	Criteria for denoting a constraint to be feasible in determining convergence in ACTCON. If maximum constraint value < CTLMIN, the design is feasible. Set in DESIGN or in FSD.
EPS	ACTCON	Criteria used in ACTCON for selecting active constraints. All constraints with values greater than EPS will be retained. (Def = -0.10, also see NRFAC)
FMAX	GDR1, GDR2	The maximum frequency value associated with the NEIV eigenvalues computed for dynamic reduction in the current boundary condition.
MOVLIM	DESIGN, MAKDFV, TCEVAL	A move limit applied to the physical design variable (V) for mathematical programming methods. The move is: $V/\text{MOVLIM} < V < V * \text{MOVLIM}$ . (Def = 2.0, MOVLIM must be > 1.0)
NRFAC	ACTCON	Criteria used in ACTCON for selecting active constraints. At least NRFAC times NDV (see Table 5) constraints will be retained. (Def = 3.0, also see EPS)
QDP	BDCASE, others	Dynamic pressure value used in the current steady aeroelastic subcase. Output from BDCASE used subsequently in MAPOL expressions and modules.

TABLE 5. INTEGER SCALAR PARAMETERS IN THE STANDARD SEQUENCE

NOTE: All logic flags are zero if "false" and non-zero (either equal to 1 or a counter) if "true."

NAME	MODULES	DESCRIPTION
AAC	ABOUND	Logic flag equal to one if there are active aeroelastic constraints in the active boundary condition.
ABC	ABOUND	Logic flag equal to one if the current boundary condition is "active" for sensitivity analysis.
ADC	ABOUND, FREQSENS	Logic flag equal to the number of active frequency constraints in the current active boundary condition.
AFC	ABOUND, FLUTSENS	Logic flag equal to the number of active flutter constraints in the current active boundary condition.
BBLAST	BDCASE	Logic flag equal to one if there are any nuclear blast subcases in the current boundary condition.
BC	N/A	Boundary condition loop counter.
BDFR	BDCASE	Logic flag equal to one if there are any direct frequency response subcases in the current boundary condition.
BDRSP	BDCASE	Logic flag equal to one if there are either transient or frequency response disciplines in the current boundary condition.
BDTR	BDCASE	Logic flag equal to one if there are any direct transient response subcases in the current boundary condition.
BDYN	BDCASE	Logic flag equal to one if there are any dynamic analyses (flutter, transient or frequency) in the current boundary condition.
BFLUTR	BDCASE	Logic flag equal to one if there are any flutter analyses in the current boundary condition.
BGUST	BDCASE	Logic flag equal to one if there are any gust loads for either transient or frequency disciplines in the current boundary condition.

TABLE 5. INTEGER SCALAR PARAMETERS IN THE STANDARD SEQUENCE (Continued)

NOTE: All logic flags are zero if "false" and non-zero (either equal to 1 or a counter) if "true."

NAME	MODULES	DESCRIPTION
BLOAD	BDCASE	Logic flag equal to one if there are any mechanical, thermal or gravity static applied loads in the current boundary condition.
BMASS	BDCASE	Logic flag equal to one if the disciplines within the boundary condition require reduction of the mass matrix.
BMFR	BDCASE	Logic flag equal to one if there are any modal frequency response subcases in the current boundary condition.
BMODES	BDCASE	Logic flag equal to one if there are disciplines that require that a normal modes analysis be performed.
BMTR	BDCASE	Logic flag equal to one if there are any modal transient response subcases in the current boundary condition.
BSAERO	BDCASE	Logic flag equal to one if there are any steady aeroelastic subcases in the current boundary condition.
CONVERGE	ACTCON, DESIGN, FSD	Global optimization convergence flag. If zero, then not converged; if one, then the approximate problem converged; if two, then global convergence has been reached.
DDFLG	DDLLOAD	Logic flag equal to one if the current statics subcases contain design dependent (gravity or thermal) loads.
FSDFLG	FSD	Logic flag equal to one if the current iteration is to be performed using the fully stressed design option.
GNORM	GDR3	The sum of LJSET and LKSET.
GSIZE	IFP, others	The number of structural degrees of freedom in the model. Output from IFP and subsequently used in many modules.
GSIZEN	GDR4	"GSIZE" modified subject to dynamic reduction.

TABLE 5. INTEGER SCALAR PARAMETERS IN THE STANDARD SEQUENCE (Continued)

NOTE: All logic flags are zero if "false" and non-zero (either equal to 1 or a counter) if "true."

NAME	MODULES	DESCRIPTION
HSIZE	FLUTTRAN, OFPEDR, REIG	Number of eigenvectors extracted by the REIG module.
LJSET	GDR1, GDR2, GDR3, GDR4	Number of degrees of freedom in the j-set in dynamic reduction.
LKSET	GDR1, GDR2, GDR3, GDR4	Number of degrees of freedom in the k-set in dynamic reduction.
MAXFSD	FSD	Parameter set in the MAPOL sequence indicating the number of leading FSD cycles to be performed if FSD is selected. (Def=3, MAXFSD <= MAXITER)
MAXITER	ACTCON	Parameter set in the MAPOL sequence indicating the maximum number of resizing cycles that are to be performed. (Def = 15)
MINDEX	ABOUND, AEROSENS, BDCASE, PFAERO	The index value for the Mach number dependent subscripted steady aerodynamic matrices. Typically has a value used to select the proper matrices for the current boundary condition.
NACSD	ABOUND	The number of active stress and displacement constraints in the current active boundary condition. Used to select either the virtual load or gradient method in sensitivity analysis.
NAE	ABOUND, AEROSENS	The number of aeroelastic effectiveness constraints in the current active boundary condition.
NAERO	PFAERO	A looping counter in PFAERO that is set on the first pass. It is equal to the number of distinct Mach numbers appearing in aeroelastic trim conditions in the Bulk Data packet.
NAU	ABOUND, AEROSENS	The number of active static load conditions in the current active boundary condition. Initially includes pseudo-loads for aeroelastic effectiveness constraints. They are then subtracted in the AEROSENS module.



TABLE 5. INTEGER SCALAR PARAMETERS IN THE STANDARD SEQUENCE (Continued)

NOTE: All logic flags are zero if "false" and non-zero (either equal to 1 or a counter) if "true."

NAME	MODULES	DESCRIPTION
NAUE	ABOUND, AEROSENS	The number of active pseudo-load conditions associated with active aeroelastic effectiveness constraints.
NBNDCOND	SOLUTION	The total number of boundary conditions in the solution control packet. Equal to the number of optimization boundary conditions plus the number of analysis boundary conditions.
NDV	MAKEST, others	The number of global design variables in the design model. Set by MAKEST and used in many subsequent modules.
NEG1	N/A	A variable having the value of negative one (-1). Used in FBS and GFBS large matrix utilities.
NEIV	GDR1, GDR2	An output from GDR1 indicating the number of eigenvalues below the maximum frequency specified for dynamic reduction.
NGDR	BOUND	Logic flag equal to negative one if dynamic reduction is selected for the current boundary condition.
NITER	N/A	The current optimization iteration number.
NMPC	BOUND, ABOUND	Logic flag equal to the number of degrees of freedom in the multipoint constraint set for the current boundary condition.
NOMIT	BOUND, ABOUND	Logic flag equal to the number of omitted degrees of freedom in the current boundary condition.
NRSET	BOUND, ABOUND	Logic flag equal to the number of support degrees of freedom in the current boundary condition.
NSPC	BOUND, ABOUND	Logic flag equal to the number of single point constraint degrees of freedom in the current boundary condition.

TABLE 5. INTEGER SCALAR PARAMETERS IN THE STANDARD SEQUENCE (Concluded)

NOTE: All logic flags are zero if "false" and non-zero (either equal to 1 or a counter) if "true."

NAME	MODULES	DESCRIPTION
NUMOPTBC	SOLUTION	The number of optimization boundary conditions in the solution control packet.
OPSTRAT	DESIGN, FSD, SOLUTION	The optimization strategy selected in the solution control.
SYM	BDCASE	A control flag denoting whether the symmetric (SYM=1) or antisymmetric (SYM=-1) steady aeroelastic matrices are to be used in the current boundary condition.
TRMTYP	ABOUND, BDCASE, others	A control flag denoting the type of trim selected for the current steady aeroelastic subcases. Output from either BDCASE (analysis) or ABOUND (sensitivity) and used in many subsequent modules. TRMTYP = 0    no trim TRMTYP = 1    lift only trim TRMTYP = 2    lift/pitch trim

#### 4.4.2 The Solution Algorithm

Finite element analysis, which forms the core of the ASTROS system, requires the manipulation of large matrices. The MAPOL control language was designed with this requirement in mind and, therefore, is able to directly support the manipulation of matrices. Consequently, the majority of the MAPOL sequence consists of matrix equations. The algorithmic nature of the MAPOL syntax allows the reader to follow these matrix operations fairly easily, and the notation roughly follows that used in the Theoretical Manual. Therefore, the focus of this section will be the description of modules called by the MAPOL sequence.

There are a number of engineering and utility modules called to perform tasks associated with the several analysis disciplines supported by the ASTROS system. Section 4.4.2.1 lists the modules defined to the ASTROS executive system and provides a brief description of each. Not all of these

modules appear in the standard solution sequence, but are included in the table to ensure its completeness and usefulness in modifying the standard sequence. The use of these modules is discussed in more detail in the section on modifying the standard MAPOL sequence and are more fully documented in the Programmer's Manual. Brief descriptions of the remaining segments of the standard algorithm follow. Coupled with the inherent readability of MAPOL syntax, they provide a reasonably complete picture of the flow through the standard sequence.

#### 4.4.2.1 MAPOL Engineering and Utility Modules

This section contains a brief description of each of the MAPOL addressable modules defined to the ASTROS executive system. The intrinsic mathematical functions of the MAPOL language are not included. In some cases, the MAPOL calling list for particular modules has optional arguments for extra printed output or other special actions. Where these optional arguments do not appear in the standard sequence, the arguments are shown in lower case letters.

##### MAPOL ENGINEERING MODULE:    ABOUND

Purpose:                Generates flags for the current boundary condition during the sensitivity calculation. These are then returned to the executive sequence to direct the execution of the required sensitivity analyses.

##### Calling Sequence:

```
CALL ABOUND ( BC, ABC, PCA, NAU, NACSD, [PGA], ADC, AFC, AAC,  
             TRMTYP, MINDEX, NAE, NAUE, PAE, NMPC, NSPC, NOMIT,  
             NRSET, NGDR, [UG(BC)] );
```

##### MAPOL ENGINEERING MODULE:    ACTCON

Purpose:                Determines whether the design task has converged. If the optimization has not converged, this module selects which constraints are to be included in the current redesign. On termination or print request, this routine computes the values of the local design variables.

##### Calling Sequence:

```
CALL ACTCON ( NITER, MAXITER, NRFAC, NDV, EPS, CONVERGE, CTL,  
             CTLMIN, [AMAT] );
```

**MAPOL ENGINEERING MODULE: AEROSENS**

**Purpose:** Computes the sensitivities of active strength constraints and/or aeroelastic effectiveness constraints for active steady aeroelastic optimization boundary conditions.

**Calling Sequence:**

CALL AEROSENS (BC, TRMTYP, MINDEX, NDV, NAU, NAE, NAUE, PAE, [DK1V], [DRHS], [K1112(BC)], [K21(BC)], [RHS(BC)], [LHS(BC)], [PAR(BC)], [DUAV], [PGA], [DDEL DV], [AMAT] );

**MAPOL ENGINEERING MODULE: AMP**

**Purpose:** Computes the discipline dependent unsteady aerodynamic matrices for flutter, gust and blast analyses.

**Calling Sequence:**

CALL AMP ( [AJJTL], [D1JK], [D2JK], [SKJ], [QKKL], [QKJL], [QJL], [ajjdc], print );

**MAPOL ENGINEERING MODULE: BDCASE**

**Purpose:** Generates flags for the current boundary condition during the analysis phase. These are then returned to the executive sequence to direct the execution of the required analyses.

**Calling Sequence:**

CALL BDCASE ( BC, BLOAD, BMASS, BMODES, BSAERO, QDP, MINDEX, SYM, TRMTYP, BFLUTR, BDYN, BDRSP, BDTR, BMTR, BDFR, BMFR, BGUST, BBLAST );

**MAPOL ENGINEERING MODULE: BLASTDRV**

**Purpose:** Performs the response of an aircraft to a nuclear blast. Computes modal displacements, velocities and accelerations.

**Calling Sequence:**

CALL BLASTDRV ( BC, [GENM], [GENK], [GENFA], [GENQL], [DEL B], [URDB], [DWNWSH], [SLPMOD], [ELAS], [UBLASTI] );

**MAPOL ENGINEERING MODULE: BLASTFIT**

**Purpose:** Computes the interpolated time domain steady state and time dependent unsteady aerodynamic influence coefficients for blast analyses.

**Calling Sequence:**

CALL BLASTFIT ( BC, [QJL], [MATTR], [MATSS] )

MAPOL ENGINEERING MODULE: BLASTRIM

Purpose: Performs a trim analysis of an aircraft in order to establish initial conditions for a nuclear blast response calculation.

Calling Sequence:

CALL BLASTRIM ( BC, [DELM], [MRR(BC)], [URDB], [DELB] );

MAPOL ENGINEERING MODULE: BOUND

Purpose: Returns flags to the MAPOL sequence that define the matrix reduction path for the current boundary condition.

Calling Sequence:

CALL BOUND (BC, NMPC, NSPC, NOMIT, NRSET, NGDR );

MAPOL ENGINEERING MODULE: DCEVAL

Purpose: Evaluates displacement constraints in the current boundary condition.

Calling Sequence:

CALL DCEVAL ( BC, [UG(BC)] );

MAPOL ENGINEERING MODULE: DDLOAD

Purpose: Computes the sensitivities of design dependent loads for active boundary conditions.

Calling Sequence:

CALL DDLOAD ( NDV, GSIZE, BC, DDFLG, [PGA], [DPVJ] );

MAPOL ENGINEERING MODULE: DESIGN

Purpose: Performs redesign by math programming methods based on the current set of active constraints and constraint sensitivities.

Calling Sequence:

CALL DESIGN ( CONVERGE, MOVLIM, CNVRGLIM, CTL, CTLMIN, OPSTRAT, NUMOPTBC, [AMAT], print );

**MAPOL ENGINEERING MODULE: DMA**

**Purpose:** Assembles the direct and/or modal stiffness, mass and/or damping matrices including extra point degrees of freedom for dynamic analysis disciplines.

**Calling Sequence:**

```
CALL DMA ( BC, GSIZE, [MAA], [KAA], [TMN(BC)], [GSUBO(BC)],  
          NGDR, LAMBDA, [PHIA], [MDD], [BDD], [KDDT], [KDDF],  
          [MHH(BC)], [BHH], [KHHT], [KHHF(BC)] );
```

**MAPOL ENGINEERING MODULE: DYNLOAD**

**Purpose:** Assembles the direct and/or modal time and/or frequency dependent loads including extra point degrees of freedom for dynamic response disciplines.

**Calling Sequence:**

```
CALL DYNLOAD ( BC, GSIZE, [TMN(BC)], [GSUBO(BC)], NGDR, [PHIA],  
              [QHJL], [PDT], [PDF] );
```

**MAPOL ENGINEERING MODULE: DYNRSP**

**Purpose:** Computes the direct or modal displacements, velocities and accelerations for transient and frequency analyses.

**Calling Sequence:**

```
CALL DYNRSP (BC, [MDD], [BDD], [KDDT], [KDDF], [MHH(BC)], [BHH],  
            [KHHT], [KHHF(BC)], [PDT], [PDF], [QHHL(BC)], [UTRANA],  
            [UFREQA], [UTRANI], [UFREQI], [UTRANE], [UFREQE] );
```

**MAPOL ENGINEERING MODULE: EDR**

**Purpose:** Computes the stresses, strains, grid point forces and strain energies for elements selected for output for the current boundary condition.

**Calling Sequence:**

```
CALL EDR ( NUMOPTBC, BC, NITER, NDV, GSIZE, EOSUMMARY, EODISC,  
          [UG(BC)], [UG(BC)], [blug], [UTRANG], [UFREQG],  
          [PHIG(BC)] );
```

MAPOL ENGINEERING MODULE: EMA1

Purpose: Assembles the element stiffness and mass matrices (stored in the KELM and MELM entities) into the design sensitivity matrices DKVI, DMVI.

Calling Sequence:

CALL EMA1 ( NDV, GMKCT, DKVI, GMMCT, DMVI );

MAPOL ENGINEERING MODULE: EMA2

Purpose: Assembles the element stiffness and mass matrix sensitivities (stored in the DKVI and DMVI entities) into the global stiffness and mass matrices for the current design iteration.

Calling Sequence:

CALL EMA2 ( GSIZE, [KGG], [MGG], niter );

MAPOL ENGINEERING MODULE: EMG

Purpose: Computes the element stiffness, mass, thermal load and stress component sensitivities for all structural elements.

Calling Sequence:

CALL EMG ( NDV, GSIZE );

MAPOL UTILITY MODULE: EXIT

Purpose: Terminates the execution of the MAPOL sequence. Useful to terminate modified MAPOL sequences.

Calling Sequence:

CALL EXIT;

MAPOL ENGINEERING MODULE: FCEVAL

Purpose: Evaluates the current value of all frequency constraints.

Calling Sequence:

CALL FCEVAL ( BC, LAMBDA );

MAPOL ENGINEERING MODULE: FLUTSENS

Purpose: Computes the sensitivities of active flutter constraints in the current active boundary condition.

Calling Sequence:

CALL FLUTSENS ( BC, GSIZE, NDV, [QHHL(BC)], [MHH(BC)], [KHHF(BC)],  
[PHIG(BC)], [AMAT] );

MAPOL ENGINEERING MODULE: FLUTTRAN

Purpose: Performs flutter analyses in the current boundary condition and evaluates any flutter constraints if it is an optimization boundary condition with applied flutter constraints.

Calling Sequence:

CALL FLUTTRAN ( BC, [QHHL(BC)], LAMBDA, HSIZE, [MHH(BC)], [KHHF(BC)],  
print );

MAPOL ENGINEERING MODULE: FREDUCE

Purpose: Reduces the f-set stiffness, mass and/or loads matrix to the a-set if there are omitted degrees of freedom.

Calling Sequence:

CALL FREDUCE ( [KFF], [PF], [PFOA(BC)], bsaero, [KOOINV(BC)],  
[koou(bc)], [kao(bc)], [GSUBO(BC)], [KAA], [PA], [PO] );

MAPOL ENGINEERING MODULE: FREQSENS

Purpose: Computes the sensitivities of active frequency constraints in the current active boundary condition.

Calling Sequence:

CALL FREQSENS ( BC, ADC, NDV, [PHIG(BC)], [AMAT] );

MAPOL ENGINEERING MODULE: FSD

Purpose: Performs redesign by a fully stressed design method (Stress Ratio Algorithm) based on the set of applied stress constraints. All other applied constraints are ignored.

Calling Sequence:

CALL FSD ( NDV, NITER, MAXFSD, OPSTRAT, ALPHA, FSDFLG, CNVRGLIM,  
CONVERGE, CTL, CTLMIN );



MAPOL ENGINEERING MODULE: GDR1

Purpose: Computes the shifted stiffness matrix and the rigid body transformation matrix [GGO] to be used in Phase 2 of Generalized Dynamic Reduction.

Calling Sequence:

CALL GDR1 ( [KOO], [MOO], [KSOO], [GGO], LKSET, LJSET, NEIV, NMAX, BC );

MAPOL ENGINEERING MODULE: GDR2

Purpose: Computes the orthogonal basis [PHIOK] for the general Krylov subspace to be used in Phase 3 of Generalized Dynamic Reduction.

Calling Sequence:

CALL GRD2 ([LSOO], [MOO], [PHIOK], LKSET, LJSET, NEIV, NMAX, BC);

MAPOL ENGINEERING MODULE: GDR3

Purpose: Computes the transformation matrix [GSUBO] for Generalized Dynamic Reduction.

Calling Sequence:

CALL GDR3 ( [KOO], [KOA], [MGG], [PHIOK], [TMN(BC)], [GGO], [PGMN(BC)], [PNSF(BC)], [PFOA(BC)], [GSUBO(BC)], LKSET, LJSET, NOMIT, NRSET, GNORM, BC );

MAPOL ENGINEERING MODULE: GDR4

Purpose: Computes transformations between displacement sets useful for data recovery from Generalized Dynamic Reduction.

Calling Sequence:

CALL GDR4 ( BC, GSIZE, GSIZEN, LKSET, LJSET, NUMOPTBC, NBNDCOND, [GDRGO(BC)], not used, not used, not used, not used, not used, [PARL(BC)] );

MAPOL ENGINEERING MODULE: GREduce

Purpose: Reduces the symmetric g-set stiffness, mass or loads matrix to the n-set if there are multipoint constraints in the boundary condition.

Calling Sequence:

CALL GREduce ( [KGG], [pg], [PGMN(BC)], [TMN(BC)], [KNN], [pn] );

MAPOL ENGINEERING MODULE: GTLOAD

Purpose: Assembles the current static applied loads matrix for any statics subcases in the current boundary condition from the constant simple load vectors and the design dependent load sensitivities.

Calling Sequence:

CALL GTLOAD ( BC, GSIZE, NLOADS, [PG] );

MAPOL ENGINEERING MODULE: IFP

Purpose: Reads the Bulk Data File and loads the input data into relations. Computes the external coordinate system transformation matrices and creates the basic grid point data.

Calling Sequence:

CALL IFP ( GSIZE, sort, echo );

MAPOL ENGINEERING MODULE: INERTIA

Purpose: Computes the rigid body accelerations for static analyses with inertia relief.

Calling Sequence:

CALL INERTIA ( [LHS(BC)], [RHS(BC)], [AR] );

MAPOL ENGINEERING MODULE: LODGEN

Purpose: Assembles the simple load vectors and simple load sensitivities for all applied loads in the Bulk Data File.

Calling Sequence:

CALL LODGEN ( GSIZE );

MAPOL ENGINEERING MODULE: MAKDFU

Purpose: Assembles the sensitivities to the displacements of active stress and displacement constraints in the current active boundary condition.

Calling Sequence:

CALL MAKDFU ( BC, GSIZE, [DFDU] );

MAPOL ENGINEERING MODULE: MAKDFV

Purpose: Assembles the sensitivities of active thickness constraints.

Calling Sequence:  
CALL MAKDFV ( NDV, [AMAT], MOVLIM );

MAPOL ENGINEERING MODULE: MAKDVU

Purpose: Multiplies the stiffness or mass design sensitivities by the active displacements or accelerations.

Calling Sequence:  
CALL MAKDVU ( NDV, [UGA], [DKUG], GMKCT, DKVI );

MAPOL ENGINEERING MODULE: MAKEST

Purpose: Generates the element summary relational entities for all structural elements. Determines the design variable linking and generates sensitivities for any thickness constraints.

Calling Sequence:  
CALL MAKEST ( NDV );

MAPOL ENGINEERING MODULE: MKAMAT

Purpose: Assembles the constraint sensitivity matrix from the sensitivity matrices formed by MAKDFU and the sensitivities of the displacements for active static load conditions in the current active boundary condition.

Calling Sequence:  
CALL MKAMAT ( [AMAT], [DFDUF], [DUFV], PCA, [PGA] );

MAPOL ENGINEERING MODULE: MKUSET

Purpose: Generates the structural set definition entity USET for each boundary condition and forms the partitioning vectors and transformation matrices used in matrix reduction.

Calling Sequence:  
CALL MKUSET (BC, GSIZE, [YS(BC)], [TMN(BC)], [PGMN(BC)],  
[PNSF(BC)], [PFOA(BC)], [PARL(BC)] );

MAPOL ENGINEERING MODULE: NREDUCE

Purpose: Reduces the symmetric n-set stiffness, mass or loads matrix to the f-set if there are single point constraints in the boundary condition.

Calling Sequence:

CALL NREDUCE ( [KNN], [PN], [PNSF(BC)], [YS(BC)], [KFF], [KFS], [KSS], [PF] );

MAPOL UTILITY MODULE: NULLMAT

Purpose: Breaks data base equivalences from previous boundary conditions.

Calling Sequence:

CALL NULLMAT ( [matrix 1], [matrix 2], ... [matrix 10] );

MAPOL ENGINEERING MODULE: OFPDISP

Purpose: Prints selected displacements, velocities and/or accelerations from any analyses in the current boundary condition.

Calling Sequence:

CALL OFPDISP ( NUMOPTBC, BC, NITER, GSIZE, [UG(BC)], [AG(BC)], TRMTYP, [UG(BC)], [AG(BC)], [blug], [blue], [UTRANG], [UTRANE], [UFREQG], [UFREQE], LAMBDA, [PHIG(BC)] );

MAPOL ENGINEERING MODULE: OFPEDR

Purpose: Prints selected element stress, strain, force and/or strain energies from any analyses in the current boundary condition.

Calling Sequence:

CALL OFPEDR ( BC, HSIZE, NITER, TRMTYP );

MAPOL ENGINEERING MODULE: OFPLOAD

Purpose: Prints selected applied external loads from any analyses in the current boundary condition.

Calling Sequence:

CALL OFPLOAD ( NUMOPTBC, BC, NITER, GSIZE, [PG], TRMTYP, QDP, [GTKG], [AIRFRC(MINDEX)], [DELTA] );

MAPOL ENGINEERING MODULE: PFAERO

Purpose: Performs preface aerodynamic processing for both steady and unsteady aerodynamics and aerostructural interpolation.

Calling Sequence:

```
CALL PFAERO (Gsize, [AICMAT(MINDEX)], [AAICMAT(MINDEX)],  
[AIRFRC(MINDEX)], MINDEX, NAERO, [GTKG], [GSTKG],  
[UGTKG], [AJJTL], [D1KJ], [D2JK], [SKJ], print );
```

MAPOL ENGINEERING MODULE: PFBULK

Purpose: Performs a number of preface operations to form additional collections of data. The following entities are created or modified in this module:

```
TEMP, TEMPD to form GRIDTEMP  
DCONDSP to form DCENT  
DLONLY to form UDLOLY  
FREQ, FREQ1  
FREQ2 to form FREQL  
DMIG to form named entity  
TF to form TFDATA  
IC to form ICdata  
TSTEP to form OTL  
SOLUTION  
CONTROL to form EOSUMMARY, EODISC and GPFELEM
```

Calling Sequence:

```
CALL PFBULK ( Gsize, EOSUMMARY, EODISC, GPFELEM );
```

MAPOL ENGINEERING MODULE: QHHLGEN

Purpose: Computes the discipline dependent unsteady aerodynamic matrices for flutter, gust and/or blast analyses in the modal structural system.

Calling Sequence:

```
CALL QHHLGEN ( BC, [QKKL], [QKJL], [UGTKA], [PHIA], [PHIKH],  
[QHHL(BC)], [QHJL] );
```

MAPOL UTILITY MODULE: RECEND

Purpose: Terminates setting conditions on a MAPOL relational access.

Calling Sequence:

```
CALL RECEND ( Relation name );
```

**MAPOL ENGINEERING MODULE:    RECOVA**

**Purpose:**                Recovers the f-set displacements or accelerations if there are omitted degrees of freedom.

**Calling Sequence:**

CALL RECOVA ( [UA], [PO], [GSUBO(BC)], nrset, [aa], [ifm(bc)],  
bsaero, [KOOINV(BC)], [koou(bc)], [PFOA(BC)], [UF] );

**MAPOL ENGINEERING MODULE:    REIG**

**Purpose:**                Computes the eigenvalues and eigenvectors of the system as directed by the boundary METHOD selection.

**Calling Sequence:**

CALL REIG ( BC, [KAA], [MAA], [mrr(bc)], [d(bc)], LAMBDA, [PHIA],  
[MII], HSIZE );

**MAPOL UTILITY MODULE:    RELCND**

**Purpose:**                Sets conditions on attribute values for MAPOL retrieval of relational entities.

**Calling Sequence:**

CALL RELCND ( Relation name, Attribute name, Operator, value )

**MAPOL UTILITY MODULE:    RELADD**

**Purpose:**                Adds a tuple to an entity opened with RELUSE.

**Calling Sequence:**

CALL RELADD ( Relation name );

**MAPOL UTILITY MODULE:    RELEND**

**Purpose:**                Closes an entity opened from the MAPOL sequence using RELUSE.

**Calling Sequence:**

CALL RELEND ( Relation name );

**MAPOL UTILITY MODULE:    RELGET**

**Purpose:**                Retrieves a relational tuple into execution memory for a relation opened for use in the MAPOL sequence.

**Calling Sequence:**

CALL RELGET ( Relation name, Status );

MAPOL UTILITY MODULE: RELUPD

Purpose: Performs a relational update from execution memory of a tuple retrieved using RELGET.

Calling Sequence:

CALL RELUPD ( Relation name);

MAPOL UTILITY MODULE: RELUSE

Purpose: Opens a relational entity for access from the executive sequence.

Calling Sequence:

CALL RELUSE ( Relation name, Number of tuples, Status );

MAPOL ENGINEERING MODULE: SAERO

Purpose: Solves the trim equation for steady aeroelastic trim analyses. Computes the rigid and flexible stability coefficients for steady aeroelastic analyses and the aerodynamic effectiveness constraints for optimization.

Calling Sequence:

CALL SAERO ( BC, [LHS(BC)], [RHS(BC)], [AR], [DELTA], NRHS, [P2], [MRR(BC)] );

MAPOL ENGINEERING MODULE: SCEVAL

Purpose: Computes the stress and/or strain constraint values for the statics or steady aeroelastic trim analyses in the current boundary condition.

Calling Sequence:

CALL SCEVAL ( BC, [GLBSIG], [UG(BC)], TRMTYP, print );

MAPOL ENGINEERING MODULE: SOLUTION

Purpose: Interprets the solution control packet, forms the CASE entity and outputs certain key parameters to the executive sequence.

Calling Sequence:

CALL SOLUTION ( NUMOPTBC, NBNDCOND, OPSTRAT );

**MAPOL ENGINEERING MODULE:** TCEVAL

**Purpose:** Computes the current values of thickness constraints for this optimization iteration. Flushes certain entities for reuse in the current iteration including:

CONST, LAMBDA, CLAMBDA

**Calling Sequence:**

CALL TCEVAL ( MOVLIM );

**MAPOL UTILITY MODULE:** USETPRT

**Purpose:** Prints the structural set definition table from the USET entity for the specified boundary condition.

**Calling Sequence:**

CALL USETPRT ( BC );

**MAPOL UTILITY MODULE:** UTGPRT

**Purpose:** Prints several specific matrix entities in an interpretable form. The entities supported are:

[DKUG], [DMUG], [DPVJ], [DUG], [DPGV], [DUCV]  
and  
[DPTHVI], [DPGRVI]  
and  
[PG]  
and  
[DFDU]

**Calling Sequence:**

CALL UTGPRT ( BC, [mat1], [mat2], ... [mat10] );

**MAPOL UTILITY MODULE:** UTMPRT

**Purpose:** To print any matrix entity. There are two "methods:"  
(1) method = 0, expand each matrix column to contain one string starting with the first non-zero term and ending with the last non-zero term and ending with the last non-zero term in each column.  
(2) method > 0, print only the non-zero terms in each column.

**Calling Sequence:**

CALL UTMPRT ( method, [mat1], [mat2], ... [mat10] );



MAPOL UTILITY MODULE:    UTRPRT

Purpose:            To print any relational entity. Only the first twelve attributes are printed and character attributes must be eight characters in length or they will be ignored.

Calling Sequence:  
    CALL UTRPRT ( rel1, rel2, ... rel10 );

MAPOL UTILITY MODULE:    UTUPRT

Purpose:            To print any unstructured entity. There are four "types" supported:  
                  (1) type < 0    prints only the record length (in single precision words) of each record in the entity.  
                  (2) type = 0    prints each record using an integer format.  
                  (3) type = 1    prints each record using a real single precision format.  
                  (4) type = 2    prints each record using a double precision format.

Calling Sequence:  
    CALL UTUPRT ( UNSTRUCT, type );

MAPOL ENGINEERING MODULE:    YSMERGE

Purpose:            A special purpose merge utility for merging YS-like vectors (vectors of enforced displacements) into matrices for data recovery. If the YS argument is blank, null vectors are merged. If DYNFLG is non-zero the matrix UF is assumed to have the form of a dynamic response matrix (3 columns per subcase: displacement, velocity and acceleration).

Calling Sequence:  
    CALL YSMERGE ( [UN], [ys(bc)], [UF], [PNSF(BC)], dynflg );

4.4.2.2 The Preface Segment

In the context of optimization, it is obvious that invariant data should be computed only once and reused subsequently for each iteration. This is the underlying principle involved in the determination of which modules constitute preface modules. In each instance, the data generated are invariant with respect to the design variables.

The preface segment begins with a call to the solution control interpreter to determine the number and types of analyses to be performed. The input file processor (IFP) is then called, followed by modules to form basic data collections required for all analyses. The element connection data and element matrices are then formed as well as the simple loads and load sensitivities. If any aerodynamic analyses are requested in the solution control, the PFAERO and AMP modules are called to create the aerodynamic matrices required for the aeroelastic analysis.

#### 4.4.2.3 The Analysis/Optimization Segments

The remainder of the MAPOL algorithm consists of the optimization and analysis segments. Any particular boundary condition is either an optimization boundary condition (implying that the quantities computed in the disciplines selected in the solution control are constrained and that the structure is to be optimized subject to those constraints) or an analysis boundary condition. The design of the ASTROS system requires that all optimization boundary conditions precede any analysis boundary conditions. The analysis segment (labeled the "final analysis") is intended to follow an optimization with analyses in disciplines whose output values are not constrained but are of interest to the designer. It also provides the user with an opportunity to view additional output not desired within the optimization loop. Finally, the analysis segment can be used on a standalone basis to perform any desired analyses.

Both the optimization and analysis segments begin with a loop on the number of boundary conditions. They differ only in the respect that the analysis segment does not have calls to constraint evaluation modules and the optimization segment has a convergence test outside the analysis boundary condition loop. The first step in these loops is to assemble the global stiffness and/or mass matrices and, if needed, the global loads matrix. If the boundary condition includes a steady aerodynamic analysis, the appropriate matrices (computed in the preface segment) are retrieved for use within the analysis phase. Following these tasks, there are several BLOCK IF statements on the various dependent structural sets. In executing each block, the required matrix partitions and reductions are performed. Once the reduced matrices have been obtained for the analyses being performed within the loop,

the lowest level response quantities (e.g. displacements, eigenvalues, etc.) are computed. Following the solution, the execution proceeds through another group of dependent set BLOCK IF's to recover the solution vectors to the global set. At this point, the analysis segment is completed with calls to the output file processor modules to compute and output high level response quantities (e.g. stresses).

In the optimization phase of the optimization segment, the ACTCON module determines the status of the global convergence flag CONVERGE and, if the optimization is not complete, the redesign task is performed. Two redesign methods are supported by the standard sequence and selected through the Solution Control. If the option for Fully Stressed Design (FSD) is selected, the redesign is performed in the FSD modules. The alternative method employs mathematical programming methods that require sensitivity information. In this case, the sensitivities of the active constraints (chosen by ACTCON based on the NRFAC and EPS parameters) are computed.

The sensitivities of the active constraints which are explicit functions of the design variables are computed first. Then the second boundary condition loop within the optimization segment begins. The ABOUND routine determines the types of active constraints in each boundary condition and outputs logic flags to control the subsequent sensitivity computations. Then boundary condition dependent constraints which are explicit functions of the design variables (frequency and flutter) are computed. Next, the sensitivities of the constraints to the displacements for those constraints which are explicit functions of the displacements (stress and displacement constraints, for example) are computed using the MAKDFU module. For these types of constraints, the product of the stiffness sensitivities and the displacements and the product of the mass sensitivities and the displacements and accelerations are also computed and modified appropriately to account for design dependent loads and inertia relief. The resulting matrix is then reduced and used to solve for the sensitivities of the displacements to the design variables. This matrix is recovered to the free displacement set in a manner similar to the recovery of the outputs in the analysis phase of the optimization segment. The final module within the boundary condition loop for sensitivity evaluation

is MKAMAT. Within this module the constraint sensitivities to the design variables are formed from the product of the constraint and displacement sensitivity matrices previously obtained.

After all the active optimization boundary conditions have been processed, the DESIGN module is called. The approximate design problem is arranged for use by the optimizer and is solved using the strategy selected in the solution control. Following convergence of the approximate problem, execution returns to the top of the optimization loop and a complete reanalysis for all the boundary conditions is performed. Once completed, the ACTCON module determines if the global problem is converged and, if so, sets the global convergence flag to 2 causing the execution to pass to the top of the analysis segment.

If any analysis boundary conditions exist, they will be processed in a manner similar to the analysis phase of the optimization segment. After performing the requested final analyses (if any) the executive system terminates the ASTROS execution.

#### 4.4.3 Modifying the Standard MAPOL Sequence

The standard MAPOL sequence is provided to allow a user to run the ASTROS system without detailed knowledge of the MAPOL language or the standard sequence. There is not, however, any requirement that the standard sequence be used. The MAPOL Programmer's Manual (Appendix B) outlines the procedure for writing a valid MAPOL sequence, and any series of syntactically correct MAPOL statements may be used to direct the ASTROS procedure. All the engineering, utility and matrix manipulation modules shown in Section 4.4.2.1 are available to any MAPOL sequence used to direct the system. In addition, there are a number of intrinsic functions such as SIN and ABS that are also available. Their use is detailed in the MAPOL Programmer's Manual. The sophisticated MAPOL user is thus provided with a very flexible control language to manipulate the ASTROS system. This subsection describes simple modifications to the standard algorithm to print out additional data items, to fine tune the optimization algorithm and to restore an ASTROS analysis that was partially executed on a previous run. No set of examples, however, can possibly indicate the full range of available capabilities; the user is therefore cautioned not to be overly constrained by this discussion.

In order to avoid vast quantities of output and to limit the execution time, the standard output is kept to a minimum. Several utilities, listed in Section 4.4.2.1, can, however, be inserted in the standard sequence to output data stored on the data base. In addition, a utility has been written to print out the structural set definition table to aid in the debugging of the structural model. The UTMPRT, UTGPRT, UTRPRT and UTUPRT print utilities dump the contents of specified data base entities to the user's output file. These can be used anywhere in the MAPOL sequence after the specified entity has been filled with data. The USETPRT utility provides the user with the ability to print the structural set definition table (USET) in a format which aids in debugging the structural model -- a useful utility if the structural model is reasonably complex. These utilities provide the user with some simple tools to allow closer interaction with the data stored on the data base and to provide capability to more closely track the execution.

The print utilities provide data visibility without modifying the basic execution of the standard sequence. At a slightly more complex level, the user might desire to fine tune the optimization procedure or to track the iterations of the optimizer more closely. Tables 4 and 5 in Subsection 4.4.1 include a number of parameters which are used by ASTROS to direct the optimization. Any of these parameters can be changed by the user at any point in the MAPOL sequence. For example, the MOVLIM parameter (which is initialized in the MAPOL sequence) could be changed to a different value after the fifth iteration by placing the following statement immediately after the WHILE test on CONVERGE:

```
IF NITER > 5 MOVLIM = 1.5;
```

Obviously, the conditional testing could become as complex as the MAPOL programmer desires.

A more basic scalar parameter that the user might wish to modify is the MAXITER parameter which specifies the maximum number of times the optimization segment will be executed if the optimization problem does not reach convergence. The standard sequence uses a value of 15 for this parameter which has proved to be a good choice for many problems. A typical design model often takes several runs to debug, however, so the user might wish to

reduce MAXITER to a minimal number (like one) until the model has been verified. As will be discussed later, the ASTROS system does provide a limited restart capability so that the optimization can be continued from this point.

The parameters EPS, NRFAC, CTL, CTLMIN, and CNVRGLIM can be used to fine tune the constraint screening and global convergence criteria. The standard values provide a conservative constraint screening (that is, a large number of constraints are retained) and the convergence criteria is somewhat lenient in its definition of a violated constraint. These parameters can be modified to be more finely tuned to the particular design problem using the detailed definitions of these parameters given in the ACTCON module documentation and in the Theoretical Manual.

The brief discussion above does not begin to describe all the options open to the sophisticated ASTROS user. It does, however, outline some of the most commonly performed modifications to the standard MAPOL algorithm. The concepts described can be extended to a large number of similar changes; e.g., modifying the input dynamic pressure value within the MAPOL sequence could be done to avoid rerunning the base run of an ASTROS execution. At a more advanced level, the MAPOL relational data base entity utilities can be used to directly modify the design variable values or objective sensitivities.

As previously mentioned, the ASTROS system has a limited restart capability. It is not sophisticated, and therefore requires that the user be somewhat familiar with the details of the engineering modules. There are, however, some simple (and useful) forms of restart that can be used without learning these details. To set up for a restart, it is necessary that a run be made and the run-time data base saved. On the subsequent run(s), this data base will be used and declared to be "OLD" on the ASSIGN DATABASE entry in the ASTROS input stream. The following restrictions are strongly recommended:

- (1) No preface module (with the exception discussed below) should be executed in both the initial run and the restart run.
- (2) Terminate the initial run at the top or bottom of a loop, not in the middle.

The exception is that the Input File Processor module must always be called. Its execution is required in order to load the data base bit maps (see

CASDBMAP entity documentation in the Programmer's Manual) into memory. Note that, in a restart run, the IFP will append any BULK DATA in the input stream onto the previously existing data.

A further point must be made regarding the use of the ASTROS system in a restart mode: the scalar parameters that are used in the MAPOL sequence do not have their values stored on the data base. Therefore, if the module that defines a particular parameter is not being called in the restart MAPOL sequence, the value of the parameter must be explicitly set in the restart MAPOL sequence. As an example, suppose that all the preface modules were executed in the first run and the restart run picked up at the top of the optimization segment. The parameter NDV and all the parameters output from the SOLUTION module would need to be reset "by hand" in the restart sequence to have the values they had in the first run. Although this restart capability is limited and somewhat awkward to use, it is useful in the optimization task since it allows the user to terminate the execution at the end of the preface or the end of some number of iterations and restart from the current design. It can also be useful to reuse a data base to perform specialized operations developed by the user on the data collections resident on the data base files.

## SECTION V

### THE SOLUTION CONTROL PACKET

The solution control packet provides the means by which the user selects the optimization and analysis tasks to be performed by the ASTROS system, their order of execution and the engineering data related to each. The solution control commands are analogous in purpose to the NASTRAN Case Control commands but they are very different in form and subtly different in interpretation. Understanding the differences between ASTROS and NASTRAN in the area of solution control is fundamental in understanding multidisciplinary optimization in the ASTROS system because the solution control command structure follows directly from the ASTROS requirement to perform multidisciplinary analyses in a single run. It is considered critical that the user clearly understand the subtleties of solution control syntax and hierarchies. This section, therefore, augments the presentation of the solution control mechanics with a discussion of the design considerations that are embodied in the solution control commands. The detailed definition of all solution control commands is contained in Appendix D.

In ASTROS, the solution control is very closely linked to the structure of the standard MAPOL sequence. It may be advantageous for the beginning user to read the standard MAPOL sequence discussion in the preceding section and to study the Theoretical Manual discussion of multidisciplinary optimization before reading the remainder of this section.

The solution control packet is initiated with the keyword SOLUTION which follows the DEBUG and MAPOL packets (if present) in the input data stream. The packet is terminated when the BULK DATA packet, or the end of the input stream is encountered. The data are composed of solution control statements which can begin in any column and can extend over multiple physical records. Each statement is formed from a combination of keywords separated by blanks or commas as indicated in the detailed syntactical descriptions contained in Appendix D. Further, each command keyword can be abbreviated by the first four (or more) characters of the keyword. The solution control packet follows a prescribed hierarchy with the following levels:



## TYPE OF BOUNDARY CONDITION

### BOUNDARY CONDITION(S)

### DISCIPLINE(S)

### DISCIPLINE OPTION(S)

Each of these levels is discussed in the following subsections and compared and contrasted to their NASTRAN counterparts. In addition to these hierarchical commands, there are commands for output processing that can occur at several levels in the hierarchy. This section presents the available commands and output quantities, but the reader is referred to Section VII of this document for the in-depth presentation of ASTROS output processing. The hierarchical nature of the solution control means that each level generates a set of "defaults" to be carried over to the next lower level. These defaults are in force until they are overridden. The user must be aware of this feature, especially when requesting output at higher levels. It is possible to get print requests "by default" where they are not expected if one is not careful with the solution control hierarchy.

#### 5.1 TYPE OF BOUNDARY CONDITION

ASTROS has been designed primarily to be an automated design tool, but it can also perform analyses without doing any design. This is reflected in the division of the solution control packet into two subpackets, either of which is optional. The first, "OPTIMIZE," subpacket defines the boundary condition(s) and discipline(s) which will generate design constraints to be used in the redesign task. In defining an optimization boundary condition, the user either implicitly or explicitly specifies that constraints be applied to certain (discipline dependent) response quantities. ASTROS then considers the complete set of constraints from all disciplines in all optimization boundary conditions in the redesign task. The second, "ANALYZE," subpacket defines analyses that are to be performed on the possibly redesigned structure. The ANALYZE subpacket is intended to provide the designer with the means to obtain additional output that is not desired during the optimization phase or to perform additional analyses which were not performed in the design task. It can also be used to perform analyses on structures that are not to be designed at all. The form of the solution control packet is then:

## SOLUTION

OPTIMIZE STRATEGY - <n>

.  
Optimization Subpacket  
.

END

ANALYZE

.  
Analysis Subpacket  
.

END

If optimization is being performed, the OPTIMIZE subpacket must precede the ANALYZE subpacket. Any number of boundary conditions and/or disciplines can be performed in either subpacket.

### 5.2 BOUNDARY CONDITION

Each analysis discipline requires a set of physical boundary conditions and, in the case of unrestrained structures, a set of fictitious supports. These are defined in ASTROS in a manner very similar to that in NASTRAN; namely, through the definition of multipoint constraints (MPC), single point constraints (SPC) and support points (SUPORT). Unlike NASTRAN, however, ASTROS requires a more rigorous definition of a boundary condition. The reason for this is that the user must ensure that the system matrices at each stage of matrix reduction are uniquely defined by the boundary condition specification. For example, if the user intends to perform a normal modes analysis, a modal transient analysis and a modal flutter analysis in the same boundary condition, ASTROS requires that the modal representation of the system under analysis be the same for each discipline in the boundary condition. This requirement, which is necessary to efficiently perform multidisciplinary analysis, adds a number of additional parameters to the boundary condition definition beyond the MPC, SPC and SUPORT definitions. They include definitions to perform matrix reductions (available in NASTRAN through BULK DATA but not always selectable in the Case Control Deck) as well as selection of extra point degrees of freedom, transfer functions and other discipline dependent matrix assembly information. In NASTRAN, these data are either implicitly selected through the rigid format selection and/or bulk data

or are a "discipline option" in the case control deck. While the boundary condition definition in ASTROS appears to be very complex, it is relatively simple if one realizes that the fundamental purpose of the BOUNDARY command is to uniquely specify the system level matrices and the matrix reductions that should be performed on them.

There is one level of "boundary condition" specification which is not treated in the BOUNDARY command. It deals with symmetry options which play a restrictive role in multi-disciplinary analysis, especially for aerodynamic disciplines. The symmetry options are often limited by the nature of the structural and/or aerodynamic models that are defined in the bulk data packet. For example, if the structural model is a half model only, the user cannot specify that asymmetric structural boundary conditions be analyzed. This option is needed for the user to perform an asymmetric aeroelastic analysis with a structural half model. Unfortunately, this is not possible in ASTROS. Whenever possible, the implicit (model-defined) boundary condition specifications that existed in the NASTRAN bulk data definitions and in the interface between bulk data and solution control have been replaced with solution control dependent options. There are, however, still limitations imposed through the interactions between the model and the solution control on combining symmetric/antisymmetric and asymmetric boundary conditions within a single run.

There are 13 boundary condition specifications in ASTROS:

- |         |   |   |
|---------|---|---|
| SPC     | - | Selects single point constraints defining Degrees of Freedom (DOF's) with fixed or prescribed motion.                 |
| MPC     | - | Selects multi-point constraints defining dependency relations among specific DOF's.                                   |
| REDUCE  | - | Defines the DOF's to be retained after a Guyan reduction.   |
| SUPPORT | - | Defines DOF's to provide support conditions for free-free modal extraction, inertial relief and aeroelastic analyses. |
| METHOD  | - | Specifies an EIGR bulk data entry which gives eigenvalue extraction data if an eigenanalysis is to be performed.      |
| K2PP    | - | Specifies an input stiffness matrix.  |
| M2PP    | - | Specifies an input mass matrix.   |

- B2PP - Specifies an input damping matrix.
- DYNRED - Invokes dynamic reduction.
- INERTIA - Specifies a JSET bulk data set for dynamic reduction.
- ESET - Specifies the extra point DOF's to be included in dynamic response analyses.
- TFL - Specifies transfer functions that are to be included in dynamic response analyses.
- DAMPING - Specifies structural or viscous damping to be used in dynamic response analyses.

A boundary condition is defined by the BOUNDARY request and one or more of these further specifications, each of which points to bulk data entries.

As indicated above, the specification of METHOD, K2PP, M2PP, B2PP, ESET, DAMPING and TFL at this level in the hierarchy is in recognition of the fact that a number of the disciplines could require different sets of data for the associated items and it is desirable to group together operations with one set of items. This does, by definition, create a restriction that only one eigenanalysis and only one set of p-size matrices and input transfer functions can be accommodated per boundary condition. Examples of boundary definitions are:

BOUNDARY SPC = 100

BOUNDARY MPC = 10, SPC = 100

BOUNDARY SPC = 10, MPC = 20, REDUCE = 30, SUPPORT = 40

BOUNDARY SPC = 10, REDUCE = 20, METHOD = 100

BOUNDARY SPC = 1, K2PP = STIFF, M2PP = MASS, B2PP = DAMP

BOUNDARY SPC = 4, DYNRED = 2, INERTIA = 4

Note that all desired specifications are listed and that their order of appearance is not important.

Several boundary conditions may appear within a given subpacket. For example:

ANALYZE

BOUNDARY SPC = 10

STATICS

.  
.  
.

BOUNDARY SPC = 20, REDUCE = 3-, METHOD = 1111

STATICS

.  
.  
.

MODES

.  
.  
.

END

In this case, a statics analysis is performed using the first boundary condition followed by a statics and modes analysis for the second boundary condition. Note that unlike NASTRAN, the sets of points to be retained in the Guyan reduction and used for the support definition are selected.

The appearance of a BOUNDARY command leads to expensive matrix partitioning and decomposition operations. Therefore, some thought should be expended to avoid unnecessary computer resource use. For example, suppose an ASTROS execution was directed to perform static analyses with two boundaries:

SPC - 10 and SPC - 20, and a dynamic analysis with two boundaries: SPC - 10 and SPC - 100. The direct solution sequence could be:

ANALYZE

BOUNDARY SPC - 10

STATICS

BOUNDARY SPC - 20

STATICS

.  
.  
.

BOUNDARY SPC - 10, METHOD - 30

MODES

.  
.  
.

BOUNDARY SPC - 100, METHOD - 40

MODES

.  
.  
.

END

This sequence would cause four separate partitionings of the system level matrices. On the other hand, the sequence:

ANALYZE

BOUNDARY SPC = 10, METHOD = 30

STATICS

.  
.  
.

MODES

BOUNDARY SPC = 20

STATICS

.  
.  
.

BOUNDARY SPC = 100, METHOD = 40

MODES

.  
.  
.

END

eliminates one of the four partitioning operations.

### 5.3 DISCIPLINES

A number of types of analyses, or disciplines, can be performed during a given ANALYZE or OPTIMIZE boundary condition. In fact, it is this multidisciplinary capability that makes the ASTROS code viable in a preliminary design context. The preceding sections have already alluded to the fact that each of these disciplines has an associated set of commands:

## ANALYZE

BOUNDARY SPC = 30

DISCIPLINE 1

.  
.  
.

DISCIPLINE 2

.  
.  
.

END

A suite of seven disciplines are available in ASTROS. These are:

STATICS	-	Static structural analysis
MODES	-	Normal modes of vibration
SAERO	-	Steady-state aeroelastic analysis
FLUTTER	-	Aeroelastic stability analysis
TRANSIENT	-	Transient response analysis
FREQUENCY	-	Frequency response analysis
BLAST	-	Transient response to a nuclear blast.

Of these options, TRANSIENT, FREQUENCY, and BLAST do not generate any design constraints and so are not useful in OPTIMIZE boundary conditions. Should the user wish to see output from these disciplines during the optimization, however, they are supported in the OPTIMIZE subpacket.

The standard MAPOL sequence contains the restriction that the SAERO discipline cannot share its boundary condition with another discipline and two or more SAERO disciplines in the same boundary condition must use the same symmetry, Mach number and dynamic pressure. These restrictions follow from the requirement that system level matrices be unique within a boundary condition. The asymmetric aeroelastic correction matrix that is applied to the stiffness matrix precludes the combination of SAERO and non-SAERO disciplines in the same boundary condition. Further, the aeroelastic correction is a function of symmetry, Mach number and dynamic pressure, accounting for the latter restriction.



#### 5.4 DISCIPLINE OPTIONS

Each of the disciplines requires further options to completely define the execution process. These options point to set IDs in the bulk data packet that define engineering data. For example, the STATICS discipline requires that loads information be supplied. This is implemented in ASTROS by a parenthetical "phrase" attached to the STATICS discipline:

SOLUTION

OPTIMIZE STRATEGY = 57

.  
.  
.

STATICS (MECH = 10)

.  
.  
.

END

In this case, bulk data applied load entries with a set ID of 10 are used to construct a mechanical load vector in a statics analysis. In general, the discipline commands have the form:

<disc> <type> {(<option> = <n>, <option> = <n>)}

The discipline options that are available are:

MECHANICAL GRAVITY THERMAL	-	Specify load set IDs for the STATIC discipline.
TRIM	-	Specifies a TRIM bulk data entry which gives flight condition information for the SAERO discipline.
DCONSTRAINT	-	Specifies the set IDs of constraint bulk data entries that apply for the given discipline.
DLOAD	-	Specifies applied loads for the TRANSIENT and FREQUENCY disciplines.
TSTEP	-	Specifies the time step for the TRANSIENT and BLAST disciplines as well as for the discrete form of the GUST discipline option.

FSTEP	-	Specifies the frequencies for the FREQUENCY and the harmonic form of the GUST discipline option.
IC	-	Specifies the initial conditions that are to be used in the direct method for the TRANSIENT discipline.
FFT	-	Specifies that the Fast Fourier technique is to be used in the TRANSIENT or GUST disciplines.
BLCOND	-	Specifies parameters for the BLAST discipline.
FLCOND	-	Specifies parameters for the FLUTTER discipline.
GUST	-	Specifies that a gust analysis is to be performed for the accompanying transient or frequency discipline.

The discipline types are:

DIRECT	-	Specifies that the direct method is to be used in the TRANSIENT or FREQUENCY disciplines.
MODAL	-	Specifies that the modal method is to be used in the TRANSIENT or FREQUENCY disciplines.

Table 6 presents a matrix that defines options and types available for each of the disciplines. In addition, disciplines requiring particular boundary condition specifications are noted; for example, modal disciplines require a METHOD specification on the BOUNDARY command. The following subsections present each discipline in turn to more explicitly define the discipline options. Most importantly, these subsections present the definition of a "subcase" of the discipline as it is defined in the ASTROS system and present the response quantities that can be constrained in the optimization task.

#### 5.4.1 "STATICS" Discipline Options

One or more of the MECHANICAL, GRAVITY or THERMAL load specifications must be called out as a discipline option for STATICS. Each STATICS discipline constitutes one subcase (one load vector) so specifying a combination of load types will generate a linear combination of the selected loads. A reference to the LOAD bulk data entry as a MECHANICAL load can also be used to obtain linear load combinations. If the STATICS discipline appears in the OPTIMIZE subpacket, the DCONSTRAINT option can be used to refer to DCONDSP bulk data entries to apply displacement constraints. Stress constraints defined on DCONSTR bulk data entries are applied to all STATICS (and some

TABLE 6. DISCIPLINE OPTIONS MATRIX

OPTION	DISCIPLINE						
	STAT	MODE	SAER	FLUT	TRAN	FREQ	BLAS
MECH	0	---	---	---	---	---	---
GRAV	0	---	---	---	---	---	---
THERM	0	---	---	---	---	---	---
TRIM	---	---	R	---	---	---	---
DCONSTRAINT	0	0	0	0	---	---	---
DLOAD	---	---	---	---	R	R	---
TSTEP	---	---	---	---	R	---	R
FSTEP	---	---	---	---	---	R	---
IC	---	---	---	---	0	---	---
FFT	---	---	---	---	0	0	---
DIRECT	---	---	---	---	0	0	---
MODAL	---	---	---	---	0	0	---
BLCOND	---	---	---	---	---	---	R
FLCOND	---	---	---	R	---	---	---
GUST	---	---	---	---	0	0	---

- NOTES: 1. R - Required, 0 - Optional, --- - Not applicable.
2. Either DIRECT or MODAL must be specified for the TRANSIENT and FREQUENCY disciplines.
3. The BOUNDARY solution command must include a method specification if the MODES, FLUTTER or BLAST discipline is specified or if the MODAL option of TRANSIENT or FREQUENCY discipline is specified.
4. At least one of MECH, GRAV or THERM must be specified for statics.

SAERO) subcases in the OPTIMIZE subpacket if any DCONSTR are in the BULK DATA packet. Therefore DCONSTR entries are not explicitly referenced in the solution control.

#### 5.4.2 "MODES" Discipline Options

MODES is completely defined for analysis by the METHOD boundary specification, which refers to an EIGR bulk data entry selecting the eigenvalue extraction method. If, however, the modal analysis is performed in the OPTIMIZE subpacket, the DCONSTRAINT option can be used to apply frequency constraints through the DCONFRQ bulk data entry. Note that more than one frequency can be constrained and than more than one constraint can be placed on the same modal frequency. The user is warned against defining the frequency constraints in such a way as to specify an excluded range of frequencies for a mode; for example, requiring that a modal frequency be below 10 Hz OR above 20 Hz. ASTROS treats all applied constraints as Boolean AND statements so the above example would be interpreted by ASTROS as an inconsistent requirement that the frequency be both above 20 Hz AND below 10 Hz.

In ASTROS, each eigenvector is considered to be a separate subcase. It is important to note in this case that more than one subcase is represented by a single solution control discipline statement. In output requests, therefore, the subcases for which output is desired must be explicitly selected. This is presented in greater detail in Subsection 5.5 and in Section VII.

#### 5.4.3 "SAERO" Discipline Options

The SAERO discipline must have a TRIM condition specified in the solution control. For analysis, this selection completes the specification of the discipline with each TRIM condition generating up to one subcase. If the trim type on the TRIM bulk data is zero, no displacement fields and, therefore, no subcases, are generated. In the OPTIMIZE subpacket, the DCONSTRAINT option can be used to select a number of different constraint types which depend on the type of TRIM analysis selected. In general the DCONSTRAINT can refer to DCONDSP bulk data entries for displacement constraints, DCONCLA for lift effectiveness constraints and DCONALE for aileron effectiveness constraints (DCONSTR constraints are implicitly selected as noted in the STATICS subsection). The strength constraints are only applicable when the trim type selected on the TRIM bulk data entry is 1 or 2. The aileron effectiveness

constraints are only applicable for antisymmetric analyses of trim type 0 and the lift effectiveness constraints require a symmetric trim analysis of type 1 or 2. The reader is referred to Section VI and Appendix E for additional information on the TRIM bulk data entry.

#### 5.4.4 "FLUTTER" Discipline Options

The FLUTTER discipline must have one or more flight conditions specified in the solution control through the FLCOND option. For analysis, this selection completes the specification of the discipline with each FLCOND condition generating up to one "subcase" (consisting of the flutter eigenvector) for each Mach number and density ratio on the FLUTTER bulk data entry if flutter occurs. In the OPTIMIZE subpacket, the "subcase" has no meaning for flutter, although more than one flutter analysis can be performed. The DCONSTRAINT option can be used to select DCONFLT bulk data entries to place a required damping limit on each of the roots extracted in the flutter analysis. The actual flutter root and eigenvector cannot be obtained in the OPTIMIZE subpacket.

#### 5.4.5 "TRANSIENT" Discipline Options

The TRANSIENT discipline must have time step and load information specified in the solution control through the TSTEP and DLOAD options. This discipline has no associated constraints and, while it is fully supported in the OPTIMIZE subpacket, it will not generate data for use in the redesign task. There are many additional options which can be selected in transient analysis. These are (1) initial conditions, which can be selected through the IC option for DIRECT transient analyses; (2) Fast Fourier Transform techniques, which are selected with the FFT option; and (3) discrete gust loads, which are applied using the GUST option. In each case, the solution control option points to a bulk data entry having the same name.

In ASTROS, each time step for which output is saved is considered to be a separate subcase. It is important to note that, like the MODES discipline, more than one subcase is represented by a single solution control discipline statement. In output requests, therefore, the subcases for which output is desired must be explicitly selected. This is presented in greater detail in Subsection 5.5 and in Section VII.

#### 5.4.6 "FREQUENCY" Discipline Options

The FREQUENCY discipline is very similar to the TRANSIENT discipline presented in the preceding subsection. Frequency step and load information are specified in the solution control through the FSTEP and DLOAD options. This discipline has no associated constraints and, while it is fully supported in the OPTIMIZE subpacket, it will not generate data for use in the redesign task. There are two additional options which can be selected in frequency response analysis. These are (1) Fast Fourier Transform techniques, which are selected with the FFT option; and (2) harmonic gust loads, which are applied using the GUST option. In each case, the solution control option points to a bulk data entry having the same name.

In ASTROS, each frequency step for which output is saved is considered to be a separate subcase. It is important to note that, like the MODES discipline, more than one subcase is represented by a single solution control discipline statement. In output requests, therefore, the subcases for which output is desired must be explicitly selected. This is presented in greater detail in Subsection 5.5 and in Section VII.

#### 5.4.7 "BLAST" Discipline Options

The BLAST discipline's solution control is very similar to that of the TRANSIENT discipline. It must have time steps specified in the solution control through the TSTEP option but, unlike the normal transient response, DLOAD is not used to generate the load. Instead, the BLCOND option is used to select a blast condition defined on a BLAST bulk data entry. As with the other dynamic response disciplines, there are no associated design constraints. These two options completely specify the BLAST analysis.

In ASTROS, each time step for which output is saved is considered to be a separate subcase. It is important to note that, like the MODES discipline, more than one subcase is represented by a single solution control discipline statement. In output requests, therefore, the subcases for which output is desired must be explicitly selected. This is presented in greater detail in Subsection 5.5 and in Section VII.

### 5.5 OUTPUT REQUESTS

Most analysis disciplines in ASTROS have response quantities (displacements, stresses, strains, etc.) computed at either node points,

structural elements or aerodynamic elements. The user can select that these results be written to the print (output) file through the PRINT command and its associated options or written to a punch file through the PUNCH command. In addition, there are a number of solution control commands that can be used to label the output. This subsection documents the PRINT and PUNCH commands and the labeling commands and discusses their use. The PRINT and PUNCH commands are identical in form and interpretation, so the PRINT command will be used to represent both commands in the following discussion. There are also many features and utilities available to the user to obtain output through modifications to the executive (MAPOL) sequence. These include direct use of MAPOL utilities, modification of print parameters in functional module calling sequences and user written procedures or modules. These output capabilities and a more complete discussion of the output processing (PRINT and PUNCH) capabilities of the ASTROS system is presented in Section VII of this manual. The reader is referred to Appendix D for the complete PRINT and PUNCH command definitions.

The PRINT and PUNCH commands have a number of options which can be separated into three groups: subcase options, response quantity options and form options. The subcase options select a set of subcases to which the PRINT command applies while the remaining options select the actual data quantities that are desired (e.g. stresses, strains, and displacements) and the form in which complex quantities are to be printed. The output selection can appear at any level and of the solution control hierarchy and will apply at that level and below until it is overridden. When more than one discipline is covered by a print request at the boundary level, ASTROS will consider only the relevant print requests for each discipline. For example, if statics and flutter are performed, the statics discipline will ignore any ROOTS requests and the flutter discipline will ignore any STRESS requests.

#### 5.5.1 Subcase Options

As explained in the preceding subsections, some disciplines have more than one "subcase" per solution control statement. Others, like STATICS and SAERO have a separate solution control statement for each subcase. When one subcase is defined per statement, the user is free to modify the print requests from subcase to subcase; for example:

## ANALYZE

BOUNDARY SPC - 10

STATICS ( MECH - 10 )

PRINT STRESS - ALL, DISP - 100

STATICS ( MECH - 20, GRAV - 100 )

PRINT DISP - ALL

specifies that stresses for all elements and displacements for nodes listed in set 100 be printed for the first subcase (mechanical loads with set identification 10) and only the displacements be printed for the next load condition. When the discipline generates more than one subcase, however, the user MUST specify the subcases to which the PRINT request applies. For example:

## ANALYZE

BOUNDARY SPC - 10, METHOD - 1000

MODES

PRINT DISP - 100, MODES - ALL

selects that displacements (eigenvectors) for all the computed mode shapes be printed. If the MODES = ALL selection was not included in the PRINT statement, the user would get NO output. The user is cautioned that the output processing in ASTROS is designed to limit output to those quantities that are explicitly selected and, therefore, the default for subcase option MODES is that no modes are selected. Whenever multiple subcases are generated by a discipline (MODES, TRANSIENT, FREQUENCY and BLAST) a subcase selection option is REQUIRED on the PRINT command in order to get any output. The subcase options in ASTROS are:

- |           |   |   |
|-----------|---|---|
| FREQUENCY | - | Selects the frequency steps of frequency response disciplines at which output is desired by referencing a FREQLIST bulk data entry.   |
| MODE      | - | Selects the eigenvectors of a normal modes discipline at which output is desired by referencing a MODELIST bulk data entry.           |
| TIME      | - | Selects the time steps of transient response and blast analyses at which output is desired by referencing a TIMELIST bulk data entry. |



### 5.5.2 Response Quantity Options

ASTROS is able to compute a number of response quantities for each discipline type. Each discipline type generates a different set of quantities so that the quantity selected by a particular keyword can sometimes change from one discipline to another. In addition, the available quantities are a sometimes a function of the boundary condition type. For example, the flutter mode shape is not available as an output from a flutter analysis performed in the OPTIMIZE subpacket. This subsection presents the available quantities, the PRINT options which select them and the limitations (if any) on their availability.

The following PRINT and PUNCH response quantity options are available:

PRESSURE	-	Selects pressures at aerodynamic boxes.
VELOCITY	-	Selects velocities at nodal points.
DISPLACEMENTS	-	Selects displacements at nodal points.
ENERGY	-	Selects strain energy at structural elements.
GPFORCE	-	Selects grid point forces at nodal points.
LOAD	-	Selects applied loads at nodal points.
SPCFORCE	-	Selects forces of single point constraint at nodal points.
STRESS	-	Selects stresses at structural elements.
ACCELERATION	-	Selects accelerations at nodal points.
STRAIN	-	Selects strains at structural elements.
ROOT	-	Selects eigenvalues for flutter and normal modes.
DESIGN	-	Selects local and global design variable values at each iteration.
DCONSTRAINT	-	Selects active constraints at each iteration.
TRIM	-	Selects trim and stability coefficients for steady aeroelastic analyses.

As in NASTRAN, stresses, strains and element forces are computed in the element coordinate system at predetermined or user selected points and nodal quantities are computed in the global coordinate system. DESIGN and

DCONSTRAINT prints are only applicable in the OPTIMIZE subpacket and, as previously stated, the DISP option for flutter analyses is only applicable in the ANALYZE subpacket. All other options are available independent of the boundary condition type. Table 7 presents a matrix of response quantity options for each discipline type, showing the applicability of each option. Any requests for quantities that do not apply to the particular discipline will be ignored by the output processor without warning.

Most options can be ALL, NONE or an integer value which select bulk data entry sets listing the items for which the response quantity is desired. For example, the STRESS option points to the ELEMLIST bulk data entity which lists the elements for which stresses are desired. The NONE option is used to override a default established through a print or punch request at a higher level in the hierarchy. The ASTROS output philosophy is similar to that of NASTRAN in that it is assumed that mistakes in the output requests should not terminate execution. If, for example, the requested structural element does not exist in the model, the output request will be ignored without any warning to the user. Other output request errors in ASTROS are treated in a similar manner, occasionally generating a warning message, but more typically resulting in no visible indication that the request was in error. Therefore the user can, in most cases, request output that does not apply to the discipline under analysis, for entities (nodes or elements) which do not exist and/or for subcases that are not defined without causing termination of the execution.

### 5.5.3 Form Options

For complex response quantities, the form option is provided to select either RECTANGULAR or POLAR form. Rectangular form gives the cartesian components of the quantity in the rectangular complex plane in which the first number represents the real component and the second number the imaginary component. Polar form gives the components in polar coordinates in which the first number represents the radial distance from the origin (the magnitude) and the second represents the angular displacement from the real coordinate axis (the phase angle). The phase angle is computed in degrees.

The form can be specified at two levels as parenthetical "phrases" attached to the print or punch statement. At the higher level, the form

TABLE 7. RESPONSE QUANTITY PRINT OPTIONS MATRIX

OPTION	STAT	MODE	SAERO	FLUT	TRANS	FREQ	BLAS
PRESS	---	---	X	---	---	---	---
VELO	---	---	---	---	X	X	X
DISP	X	X	X	X <sup>1</sup>	X	X	X
ENERGY	X	X	X	---	X	---	X
FORCE	X	X	X	---	X	---	X
GPFORCE	X	---	X	---	---	---	---
LOAD	X	---	X	---	X	X	X
SPCFORCE	X	---	X	---	---	---	---
STRESS	X	X	X	---	X	---	X
ACCEL	X <sup>2</sup>	---	X <sup>2</sup>	---	X	X	X
STRAIN	X	X	X	---	X	---	X
ROOT	---	X <sup>3</sup>	---	X <sup>3</sup>	---	---	---
TRIM	---	---	X	---	---	---	X

- NOTES: 1. Flutter displacements (mode shapes) are only available for analysis and then only if a flutter crossing is found.
2. The accelerations are available for STATICS with inertia relief and all SAERO analyses.
3. ROOTS will print real eigenvalue extraction summary data for MODES and complex eigenvalues for FLUTTER.

option generates a default for the entire print or punch statement. For example:

```
PRINT (POLAR) STRESS - ALL, STRAIN - ALL
```

requests that polar form be used for both stress and strain response quantities. In addition, the form option can be attached to the individual quantity options to override the print default. For example:

```
PRINT (POLAR) STRESS = ALL, STRAIN ( RECT ) = ALL
```

overrides the polar default for the strain output. At both levels, the default form is rectangular and any polar requests for real output quantities are ignored.

#### 5.5.4 Labeling Options

Labeling of printed output is performed through the use of three optional commands identical in form to their NASTRAN counterparts:

- TITLE        -     A title header that will appear as the first line on each page of output.
- SUBTITLE    -     A secondary header that will appear on the second line of each page of output.
- LABEL       -     A tertiary header that is typically used to identify subcase (discipline level) output.

Each of these commands can appear at any level in the solution control hierarchy and will be applied until superseded.

## SECTION VI

### THE BULK DATA PACKET

The bulk data packet provides the ASTROS system with the engineering data needed to perform the specific tasks requested by the user. It contains the model geometries for the structural model, the aerodynamic model(s) and the design model as well as the pool of data from which the solution control requests are made. Specialized information required by the analysis disciplines (e.g., Mach number and reduced frequency pairs for unsteady aerodynamic analyses) is also provided to the system through the bulk data packet. The basic input item is the bulk data entry which is directly analogous to the NASTRAN bulk data "card." In fact, NASTRAN compatible formats were chosen for the ASTROS bulk data entries whenever possible because modern structures are often analyzed using large NASTRAN finite element models having tens of thousands of lines of bulk data. Further, these large models are usually prepared using software designed specifically to generate NASTRAN models. Thus, by utilizing NASTRAN bulk data structures where possible and by using NASTRAN's bulk data style for the additional engineering data, ASTROS is highly compatible with existing NASTRAN models and with current finite element model generation methods.

Just as in NASTRAN, the bulk data packet begins with the keyword BEGIN BULK (which may be abbreviated BEGIN) and is terminated by the optional keyword ENDDATA or by the end of the input stream. The intervening bulk data entries can appear in any order. An alphabetically sorted listing of the bulk data input will be echoed to the output file unless suppressed by the user through a MAPOL sequence modification to the IFP argument list. Similar modifications to the IFP call may select that the sorted bulk data be written to the punch file for use in post-processors, be echoed unsorted or be written to both the punch file and the output file.

All the input entries are interpreted by IFP through "templates" that are defined as part of the system generation task. The templates provide for basic error checking, establish defaults and direct the placement of the raw data onto the data base. The use of templates allows additional entries to be added to the system very simply without software changes. The definition of

the templates and the means of adding new entries are documented in the Programmer's Manual. In addition, the complete listing of ASTROS bulk data templates is included in the output summary generated by the SYSGEN system generation utility during the creation of the system data base files.

On typical restart runs, the user must be sure to call the IFP module in the executive sequence (even if there is no bulk data packet) to ensure that the proper data base environment is established prior to calling any additional engineering modules. Obviously, certain user written MAPOL sequences will not require that IFP be called. On restart with a bulk data packet in the input stream, the IFP module will append the new data onto the data from the previous run(s). There is no provision for deleting existing bulk data except through MAPOL sequence modifications. This restart feature, while limited, can be useful in many instances; e.g. when additional analysis disciplines are desired or when different output requests are desired.

The remainder of this section presents the structure of the bulk data entry for ASTROS and discusses some features of the IFP module that are useful to the general user. ASTROS bulk data entries have been carefully designed to be NASTRAN compatible, so the NASTRAN User's Manual (Reference 2) has provided much of the information in the following discussion as well as having directed the design of the IFP software. The reader is also referred to the ASTROS Programmer's Manual for more information on the IFP module and for information on the addition of new bulk data entries. An additional resource is the MSC/NASTRAN Primer (Reference 3).

#### 6.1 FORMAT OF THE BULK DATA ENTRY

Each bulk data entry consists of a required "parent" line followed by a number of optional "continuation" lines. Therefore, a single bulk data entry resides on one or more lines. The bulk data line has one mnemonic field of eight characters followed by either eight data fields of eight characters or by four data fields of 16 characters and terminates with an eight character continuation field as shown in Figure 5. The data field size (either eight or 16 characters) is determined by the presence of the optional "large field marker" in the first mnemonic field of each bulk data line. The parent line begins with a character mnemonic identifying the entry followed by 4 or 8 data fields and ending with a continuation field. The continuation lines are identical except that the leading mnemonic field contains a continuation

label which is used to link it to its parent line. This structure is identical to that in NASTRAN. One important exception to NASTRAN compatibility is that ASTROS requires that the continuation lines follow continuously from the parent line although the bulk data entries themselves can be in any order. Random placement of continuations in NASTRAN is an artifact from using physical cards that were punched with the bulk data. If the card deck were dropped, the resulting random order still had to be interpretable by the code. This feature no longer needs to be supported in light of modern computer storage methods but NASTRAN compatibility dictated that similar continuation labeling be used.

A continuation line is defined for a bulk data entry that requires more than eight (or four large) data fields. The last field of the parent line is used in conjunction with the first field of the continuation line as an identifier. The parent continuation field can contain any alphanumeric entry while the first field of the continuation line contains a plus (+) as a continuation character in column 1 followed by the last 7 characters from the parent continuation label. For the parent line, the large field marker is an asterisk (\*) following the name of the entry and signifies that large data fields are to be used. For continuation lines, the asterisk used as the continuation character plays the role of the large field marker as shown in **Figure 5**. Each bulk data line must be either all narrow field or all large field, although separate lines of a single bulk data entry can have different field widths simply by using the proper field marker. This means that the same bulk data entry in wide and narrow formats are functionally identical with no need for separate templates. Unlike NASTRAN, the continuation mnemonics need not be unique among all the bulk data entries in the bulk data packet since there is no provision for randomly sorted continuations.

The input on a bulk data line can either be in fixed format, in which each item must reside within the field to which it belongs, or in free format, in which fields are separated by commas and can be positioned anywhere to the left of the column in which the fixed field would normally start. Free format input is indicated by the appearance of a comma in the first 10 characters of the input line. ASTROS requires that each line (not each bulk data entry) be either all fixed or all free format and that each free format field be separated by a comma. The NASTRAN use of a blank character as a field separator

(a) *Small Field Entry with a Small Field Continuation*

mnemonic	data		data		data		data		data		data		data		mnemonic	
NAME	←	8	→	←	8	→	←	8	→	←	8	→	←	8	→	ABC
+ BC	←	8	→	←	8	→	←	8	→	←	8	→	←	8	→	

(b) *Small Field Entry with a Large Field Continuation*

mnemonic	data		data		data		data		data		data		data		mnemonic	
NAME	←	8	→	←	8	→	←	0	→	←	8	→	←	8	→	ABC
* BC	← 16 →				← 16 →				← 16 →				← 16 →			

(c) *Large Field Entry with a Small Field Continuation*

mnemonic	data				data				data				data				mnemonic
NAME *	← 16 →				← 16 →				← 16 →				← 16 →				ABC
* BC																	DEF
+ EF	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →	← 8 →			

(d) *Large Field Entry with a Large Field Continuation*

mnemonic	data				data				data				data				mnemonic
NAME *	← 16 →				← 16 →				← 16 →				← 16 →				ABC
* BC	← 16 →				← 16 →				← 16 →				← 16 →				

Figure 5. Bulk Data Entry Format



is not supported. When free format input is used, the continuation lines can reside on the same physical line of input with the continuation labels either included or not, as in the following equivalent examples:

```
MKAERO1, 1, , 0.3, 0.5, , , , ABC , +BC, 0.01, 0.05, 0.1, 0.2
```

```
MKAERO1, 1, , 0.3, 0.5, , , , , 0.01, 0.05, 0.1, 0.2
```

In the latter case, ASTROS will automatically generate the missing continuation mnemonics. Care must be taken, however, that the first TWO data fields of the continuation line be non-blank. If not, there is an ambiguity as to whether the first continuation field constitutes a continuation label or a data field. This ambiguity causes the IFP to terminate execution with an error indicating that there is a missing continuation line. Free format input in which the parent and continuation lines are broken into separate physical lines or which explicitly include the continuation mnemonics do not suffer this limitation. Free format input is further restricted in that the break between physical lines (if needed) must occur at a break in the logical line; i.e., the split must occur between the ending continuation field on the current logical line and the continuation field of the next logical line. This means, for the preceding example, that the first example entry could be broken into two lines between the ABC and +BC fields but nowhere else. When an entry is broken into multiple physical lines, the continuation mnemonics must be supplied. Obviously, fixed format input requires continuation mnemonics for any bulk data entries having continuation lines.

## 6.2 DATA FIELD FORMATS

The interior fields of a bulk data line can contain either integer data, real data, character data or certain combinations (e.g. either integer or real data). The template for each entry defines which type(s) of data are acceptable in each field. Each data item is limited to the number of characters that fit in the length of the field; i.e., for narrow width fields no more than eight characters can be used in the data item. Unlike NASTRAN, any extra characters will spill to the next field and will result in IFP errors; i.e., there is no provision for rounding real data to fit the field size.

In order to be considered valid, the data item must first satisfy the data type requirement as specified on the template. Real numbers, including zero, must contain a decimal point, although there are a number of formats

supported. For example, the real number 3.1 may be encoded as shown or as 3.1E0, +3.1D00, 0.31E1, 3.1+0, etc. Unlike NASTRAN, however, there CANNOT be imbedded blanks anywhere in the real number and a "D" edit descriptor is treated as a single precision number until actually loaded to a double precision relational attribute. Blank fields that do not have other defaults specified on the template will be interpreted as blank characters, an integer zero or a real zero, as required. Integer values must be formed from the ten decimal digits with an optional leading plus or minus sign. Character data consist of any combination of alphanumeric characters including any digits, decimal points, etc., with no restriction that the first character be alphabetic.

### 6.3 ERROR CHECKING IN THE INPUT FILE PROCESSOR

As mentioned in the preceding subsection, the IFP module performs basic error checking to ensure that the input data are of the correct type. In addition, the templates provide for error checks that enable the IFP to check that the data satisfy particular requirements. For example, the IFP can be directed to require that a particular value be greater than zero or be one of a finite number of selections. At its most complex, the bulk data processor checks to ensure specific relationships among data on a single bulk data entry. It is important to understand, however, that no error checks occur in the IFP to ensure that references to, and interrelationships among, multiple bulk data entries are satisfied. These more complex checks occur in subsequent engineering modules. A complete description of the available template error checks and the mechanism provided to add additional error checks is presented in the Programmer's Manual. The reader may find it helpful to study this documentation since the bulk data packet and the bulk data entries are closely linked to the software in both the SYSGEN utility and the IFP module.

### 6.4 BULK DATA ENTRY SUMMARY

This section contains a summary of all the bulk data entries in the ASTROS system separated into logically related groups. The groups are composed of either model definition entries, subcase definition entries or general list entries. Appendix E contains the detailed description of each of the entries listed in this section. Each bulk data entry is labeled either "(N)ew", "(C)hanged" or "(U)nchanged" relative to its NASTRAN counterpart.

Section 6.5 discusses the differences between NASTRAN and ASTROS for those entries that have been changed or are completely different than in NASTRAN but that use the same mnemonic and serve a similar purpose.

#### 6.4.1 Aerodynamic Load Transfer

ATTACH	(N)	Rigid load transfer definition.
SET1	(U)	A structural grid point list for spline interpolation or a mode list for omitting normal modes in flutter analysis.
SET2	(U)	Structural grid point list in term of aerodynamic macroelements.
SPLINE1	(U)	Surface spline definition for out-of-plane motion.

#### 6.4.2 Applied Dynamic Loads

DLAGS	(N)	Time and phase lag definition for a spatial load.
DLOAD	(U)	Linear combination of dynamic load sets.
DLONLY	(N)	Direct definition of dynamic spatial load.
GUST	(C)	Stationary vertical gust definition.
RLOAD1	(C)	Frequency dependent dynamic load definition.
RLOAD2	(C)	Frequency dependent dynamic load definition.
TABLED1	(C)	Tabular function definition for dynamic load generation.
TLOAD1	(C)	Time dependent dynamic load definition.
TLOAD2	(C)	Time dependent dynamic load definition.

#### 6.4.3 Applied Static Loads

FORCE	(U)	Definition of a concentrated load at a grid point.
FORCE1	(U)	Definition of a concentrated load at a grid point.
GRAV	(U)	Definition of an acceleration vector for gravity loads.
LOAD	(U)	Definition of linear load combinations.
MOMENT	(U)	Definition of a moment at a grid point.
MOMENT1	(U)	Definition of a moment at a grid point.
PLOAD	(U)	Definition of a pressure load over an area.
TEMP	(U)	Definition of a temperature at a structural node.
TEMPD	(U)	Definition of default nodal temperatures.

#### 6.4.4 Boundary Condition Constraints

ASET	(C)	Analysis set definition.
ASET1	(C)	Analysis set definition.

DYNRED	(N)	Dynamic reduction parameters.
JSET	(N)	Inertia relief mode shape parameter definition.
JSET1	(N)	Inertia relief mode shape parameter definition.
MPC	(U)	Multipoint constraint definition.
MPCADD	(U)	Definition of combinations of MPC sets.
OMIT	(C)	Omit set definition.
OMIT1	(C)	Omit set definition.
SPC	(U)	Single point constraint/enforced displacement definition.
SPC1	(U)	Single point constraint definition.
SPCADD	(U)	Definition of combinations of SPC sets.
SUPPORT	(C)	Definition of coordinates for determinate reactions.

#### 6.4.5 Design Constraints

DCONALE	(N)	Aileron effectiveness constraint definition.
DCONCLA	(N)	Lift effectiveness constraint definition.
DCONDSP	(N)	Displacement constraint definition.
DCONFLT	(N)	Flutter constraint definition.
DCONFRQ	(N)	Modal frequency constraint definition.
DCONSTR	(N)	Stress/strain constraint definition.
DCONTHK	(N)	Thickness constraint definition for use with shape function design variable linking.

#### 6.4.6 Design Variables, Linking and Optimization Parameters

DESELM	(N)	Unique physical design variable definition.
DESVAR	(N)	Linked physical or shape function design variable definition.
ELIST	(N)	Shape function definition.
MPPARM	(N)	Mathematical programming default parameter override.
PLIST	(N)	Physical design variable linking definition.

#### 6.4.7 Geometry

CORD1C	(U)	Cylindrical coordinate system definition.
CORD1R	(U)	Rectangular coordinate system definition.
CORD1S	(U)	Spherical coordinate system definition.
CORD2C	(U)	Cylindrical coordinate system definition.
CORD2R	(U)	Rectangular coordinate system definition.
CORD2S	(U)	Spherical coordinate system definition.

EPOINT (C) Extra point definition for dynamics.  
 GRDSET (U) Default parameters for fields on the GRID entry.  
 GRID (U) Grid point location and coordinate system selection.  
 SPOINT (U) Scalar point definition.

#### 6.4.8 Material Properties

MAT1 (U) Isotropic elastic properties definition.  
 MAT2 (U) Two-dimensional anisotropic properties definition.  
 MAT8 (U) Orthotropic properties definition.  
 MAT9 (U) Anisotropic properties definition for isoparametric hexahedral elements.

#### 6.4.9 Miscellaneous Inputs

\$ (U) Commentary data.  
 CONVERT (N) Conversion factor definitions.  
 DMI (C) Direct matrix input.  
 DMIG (C) Direct matrix input at structural nodes.  
 MFORM (N) Mass matrix form (LUMPED or COUPLED).  
 SAVE (N) List of data base entities not to be purged.  
 SEQGP (U) Structural set resequencing definition.

#### 6.4.10 Output Selection Lists

ELEMLIST (N) List of elements for element dependent output.  
 FREQLIST (N) List of frequency steps for which output is desired.  
 GRIDLIST (N) List of nodes for nodal dependent output.  
 MODELIST (N) List of normal modes for which output is desired.  
 TIMELIST (N) List of time steps for which output is desired.

#### 6.4.11 Steady Aerodynamics

AEROS (N) Reference parameters.  
 AEFACT (N) List of real parameters.  
 AELIST (N) List of aerodynamic elements (boxes).  
 AESURF (N) Aerodynamic control surface definition.  
 AIRFOIL (N) Airfoil property definition.  
 AXSTA (N) Body axial station parameter definition.  
 BODY (N) Body configuration definition.  
 CAERO6 (N) Macroelement (panel) definition.  
 PERO6 (N) Body parameter definition.

#### 6.4.12 Structural Element Connection

BAROR	(U)	Definition of default parameters for the CBAR bar element.
CBAR	(C)	Prismatic beam element.
CELAS1	(C)	Scalar elastic spring element.
CELAS2	(C)	Scalar elastic spring element.
CIHEX1	(U)	Linear isoparametric hexahedral element.
CIHEX2	(U)	Quadratic isoparametric hexahedral element.
CIHEX3	(U)	Cubic isoparametric hexahedral element.
CMASS1	(C)	Scalar mass element.
CMASS2	(C)	Scalar mass element.
CONM1	(U)	Direct 6 x 6 mass matrix definition at a structural node.
CONM2	(C)	Concentrated mass at a structural node.
CONROD	(C)	Rod element.
CQDMEM1	(C)	Isoparametric quadrilateral membrane element.
CQUAD4	(C)	Isoparametric quadrilateral element with bending and membrane stiffness.
CROD	(C)	Rod element.
CSHEAR	(C)	Shear panel.
CTRMEM	(C)	Constant strain triangular membrane element.

#### 6.4.13 Structural Element Properties

PBAR	(C)	Prismatic beam element.
PCOMP	(C)	Composite laminate definition for CQDMEM1, CQUAD4, and CTRMEM elements.
PCOMP1	(N)	Composite laminate definition for CQUAD4 elements.
PCOMP2	(N)	Composite laminate definition for CQUAD4 elements.
PELAS	(C)	Scalar elastic spring element.
PIHEX	(U)	Linear, quadratic and cubic isoparametric hexahedral element.
PMASS	(C)	Scalar mass element.
PQDMEM1	(C)	Isoparametric quadrilateral membrane element.
PROD	(C)	Rod element.
PSHEAR	(C)	Shear panel.
PSHELL	(C)	Definition of shell element properties for CQUAD4 elements.
PTRMEM	(C)	Constant strain triangular membrane element.

#### 6.4.14 Unsteady Aerodynamics

AERO	(C)	Reference parameters.
CAERO1	(U)	Aerodynamic macroelement (panel) definition.
CAERO2	(U)	Body configuration definition.
FLFACT	(U)	Parameter definition for flutter analysis.
MKAERO1	(C)	Table of symmetries, Mach numbers, and reduced frequencies.
MKAERO2	(C)	Table of symmetries, Mach numbers, and reduced frequencies.
PAERO1	(U)	Association between bodies and macroelements.
PAERO2	(U)	Body cross-section property definition.

#### 6.4.15 Discipline Dependent Problem Control

The following bulk data entries are the controlling entries referenced by Solution Control in selecting specific disciplines and subcases. In each case, many of these inputs can appear in the bulk data packet with the particular input to be used for the subcase referenced in the Solution Control Packet.

BLAST	(N)	Parameters for nuclear blast analyses.
FLUTTER	(C)	Basic parameters for flutter analyses.
TRIM	(C)	Flight condition for steady aeroelastic trim analyses.
EIGR	(U)	Real eigenvalue extraction parameters.
FFT	(N)	Fast Fourier Transform parameter definition.
FREQ	(U)	Frequency step definition for frequency response.
FREQ1	(U)	Frequency step definition for frequency response.
FREQ2	(U)	Frequency step definition for frequency response.
IC	(N)	Initial condition definition for direct transient response (same as NASTRAN TIC card).
TABDMP1	(C)	Modal damping table for modal dynamic response.
TF	(U)	Dynamic transfer function definition.
TSTEP	(U)	Time step definition for transient response.
VSDAMP	(N)	Definition of viscous damping based on equivalent structural damping.

#### 6.5 SUMMARY OF ASTROS NASTRAN BULK DATA ENTRY DIFFERENCES

The bulk data entries marked "new" in the preceding subsection do not exist in the MSC/NASTRAN or in the COSMIC/NASTRAN versions that guided the

definition of the bulk data entries. Some of them do however, exist in other NASTRAN systems: the DYNRED, JSET, JSET1, PCOMP1, and PCOMP2 entries are examples. Others take the place of the NASTRAN "PARAM" entry which had been overused. Examples of these inputs are the CONVERT, MFORM and VSDAMP entries. The steady aeroelastic model is completely new to ASTROS since NASTRAN uses the same modeling for both steady and unsteady analysis. Also, the NASTRAN mechanism for defining dynamic loads was needlessly complicated. Working from the NASTRAN inputs, a simpler, but equally general, set of entries was developed. This resulted in the generation of a number of new entries and the modification of others. The definition of the design variables, design variable linking and the design constraints is, of course, completely new for ASTROS.

The majority of the "changed" entries have been modified to accommodate the design task. In these cases, the bulk data entry is often identical to the NASTRAN version for use in analysis with optional additional fields to specify the design data. The element connectivity and property entries are all examples of this type of change in that additional field(s) have been added to specify the maximum and minimum allowable physical design variable value if shape function design variable linking is used. In cases where data from NASTRAN preprocessors are used, there are no changes required unless shape function linking is desired.

A more subtle set of changes was required to perform multidisciplinary analysis (which, in turn, is a requirement to perform useful design optimization). In NASTRAN, as mentioned in the discussion of the Solution Control packet, many parameters were specified as part of the model definition or discipline specification because the code was limited to performing a single analysis of the given discipline. In order to remove these artificial restrictions, these data have been moved to the proper discipline's subcase definition. Examples of this form of modification are the addition of symmetry options to the MKAERO1, GUST, FLUTTER, and TRIM entries and the removal of subcase dependent data from the AERO entry. Further, the ASETi, OMITi and EPOINT entries were modified to include a set identification number to enable multiple boundary conditions and multiple control systems to be analyzed simultaneously.



The last type of modification came about because of the nature of the ASTROS data base management system. These were limited to the DMI and DMIG entries for direct loading of data base entities. The NASTRAN inputs were not compatible with the ASTROS data base and so had to be modified. In fact, these entries, while having the same name as a NASTRAN card, are completely new entries for ASTROS. A minor additional modification to the input definitions was made for the table entries (e.g., TABLEDD1 and TABDMP1) to make them more compact and to remove the useless "ENDT" table termination symbol. In ASTROS, all tabular input entries are terminated when no more data appears and do not allow a specific declaration of the table end.

While a seemingly large number of bulk data entries have been changed relative to their NASTRAN counterparts, in fact only a few have been changed in such a way that the NASTRAN version will not work in analysis. By far, the majority of the modeling bulk data entries are completely unchanged except for certain design variable linking options. In unsteady and steady aerodynamic disciplines, care must be taken to account for the subcase dependencies that NASTRAN defined implicitly or with PARAM cards. Finally, the use of ASET and OMIT entries will cause minor problems in that ASTROS requires a set identification for these entries. While this latter restriction can require some effort to fix, the gain in capability mandated that the bulk data entry be modified.

The most serious potential problem using NASTRAN models in ASTROS is that the set of bulk data entries is more limited in ASTROS than in NASTRAN. The ASTROS system has been developed primarily as a multidisciplinary preliminary design tool and does not yet contain the wide range of options supported by a mature code like NASTRAN. The many NASTRAN input entries supporting these options, therefore, have not been defined to the ASTROS system because they are not supported by any ASTROS code. Thus, there will be instances where a NASTRAN input deck will have to be modified to remove these entries which serve no purpose in ASTROS. The majority of these bulk data entries deal with unsupported elements, plotting options, output options, etc., which are not felt to present a major problem. More important is the support for NASTRAN's model definitions, most of which have already been adopted by ASTROS. One exception is the rigid elements like COSMIC NASTRAN's CRIGID1 and MSC/NASTRAN's RBAR, RBE1, and RBE2 which are not included in ASTROS. It is

anticipated that these elements may present a problem in that ASTROS only accepts multipoint constraints explicitly defined on MPC entries. In NASTRAN models using rigid elements, the user is burdened, therefore, with the task of manually translating the rigid elements into their equivalent MPC entries. There is, however, no loss of functionality in ASTROS since rigid elements are only a convenience and do not represent any capability that is not provided by the raw MPC data.

## SECTION VII

### ASTROS OUTPUT FEATURES

In a software system the magnitude of ASTROS, the amount of data that may be of interest to the user is very large. In multidisciplinary optimization, the quantity of data is even larger and the expense involved in its computation significant. It is worthwhile, therefore, to limit the amount of output to a minimum and to provide a mechanism for the user to select those data that are of importance in each particular case. Section V of this report documented one mechanism provided in ASTROS to select particular disciplines, subcases and response quantities: that of the Solution Control output request. This section endeavors to present the totality of output options available to the ASTROS user. The system controlled outputs from the engineering modules are described in order to establish a familiarity with an ASTROS output listing. This is followed by a more complete description of output from each Solution Control request than is contained in Section V, with different disciplines, elements, design constraints and node types accounted for in some detail. These represent the outputs that are fully supported by the ASTROS software and require little or no user intervention to obtain. The presentation of these features assumes that the standard executive sequence is used. If the user substantially modifies the standard sequence (to the point where certain modules are not called), some or all of the presented output features may no longer be available. Some of the documented features have not been implemented in the initial production release of the code. These features are described to the extent that they have been designed and their status at the time of initial release is noted.

The more advanced forms of user output requests are also presented in this section. The most basic of these forms involve changing the engineering module argument lists in the standard executive (MAPOL) sequence for those modules that support print level selection. For ASTROS, the defaults (established in the standard sequence) select the least printed output in each case. Finally, the MAPOL addressable print utilities are presented. The use of these utilities, in conjunction with the general versatility of the MAPOL language, provides the user with the capacity both to look at existing data and to compute and view additional data. In fact, these options enable the

user to obtain (in a crude format) virtually any data that reside on the data base or that can be computed and stored on the data base.

## 7.1 SYSTEM CONTROLLED OUTPUT

Many of the engineering and executive system program units write data to the ASTROS output listing automatically. As indicated in the introduction to this section, output of this nature in ASTROS is very limited, but sufficient amounts exist to justify a brief presentation of the data and their formats. It is also useful to present the basic ASTROS listing in order to facilitate contrasting it to listings containing user selected output quantities. The first page of ASTROS output is the title page showing the version number, date and host machine of the user's system. Each page of output following the solution control listing will be labeled with six lines of header information including the user selected title, subtitle and label. The version number, date and, if applicable, the design iteration number will also appear in the header of each page.

### 7.1.1 Default Output Printed by Modules

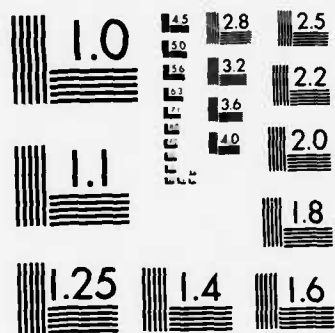
If a solution control packet is in the input data stream, the solution control commands will be echoed to the output listing following the title page. This listing is helpful in identifying the particular disciplines and cases selected in the run. The multidisciplinary nature of ASTROS requires further output labeling; therefore, in addition to the solution control summary, the EDCASE module writes a summary of selected disciplines for each boundary condition at the top of the boundary condition loop. It indicates all disciplines and most discipline options in the current boundary condition in order to assist the user in determining the particular path that will be taken through the standard MAPOL sequence. A similar print (from the ABOUND module) appears at the top of the sensitivity phase boundary condition loop to indicate the nature of the active boundary conditions and active design constraints.

Additional active constraint information is provided in the Active Constraint Summary from the ACTCON module. It indicates the total number of constraints considered active according to the current constraint deletion criteria. The user may select a complete listing of the active constraints through the PRINT DCONSTRAINT solution control option, but may not suppress

**THIS REPORT HAS BEEN DELIMITED  
AND CLEARED FOR PUBLIC RELEASE  
UNDER DOD DIRECTIVE 5200.20 AND  
NO RESTRICTIONS ARE IMPOSED UPON  
ITS USE AND DISCLOSURE.**

**DISTRIBUTION STATEMENT A**

**APPROVED FOR PUBLIC RELEASE,  
DISTRIBUTION UNLIMITED.**



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

the table header indicating the number of constraints retained of the total number applied. This number is computed even if the current design is considered to be the converged optimum. A summary of the convergence criteria and of the critical constraint value is included in the Active Constraint Summary header if the approximate problem was considered converged following the preceding redesign step.

Each redesign step is summarized in a small table entitled the Approximate Optimization Summary. It indicates the optimization method used (either fully stressed design or mathematical programming) and the change in the objective function during the solution of the approximate optimization problem. It also echoes the values of the two approximate problem termination criteria: the absolute objective function change and the percent change. The status of the (approximate problem) convergence flag following the redesign is also indicated.

The last default design print is generated by the ACTCON module on the final design iteration. The ACTCON module prints out the design iteration history, the final global design variable values and the final local design variable values. The iteration history includes statistics summarizing each approximate optimization problem and shows the increments in the objective function. The global design variable print lists the final values, the upper and lower bounds, the objective sensitivity to the design variable, the linking option used and a user label identifying the global design variable. Finally, the local variable print shows the actual physical value represented by the final design. Also included are element dependent values (e.g.,  $T/T_{MIN}$  for plate elements and inertias for BAR elements). Both the global and local design variable prints can be selected at all iterations through the Solution Control, but will be printed on the final iteration irrespective of the print selection. A more complete description of the design variable print formats is contained in Subsection 7.2.3.

The final default outputs are a trailer indicating the status of the termination (either with or without errors), the date and the time the run was completed and an execution timing summary. The timing summary indicates the CPU time spent in each phase of the execution. Each engineering and mathematical module called by the executive system is listed in the order called along with an indication of the running and step CPU times. Also, the elapsed clock

time is shown upon leaving each module. This summary is useful in determining where a problem may have occurred and in confirming that the proper path was taken through the MAPOL sequence. It is, of course, also useful as an indication of the relative CPU costs of each engineering module.

#### 7.1.2 Error Message Output

Error messages can be printed from virtually all the modules of the ASTROS system as well as from the data base management software. Data base errors should not occur unless the user has modified or otherwise written a special MAPOL sequence, incorrectly assigned file names or used other incorrect or inconsistent data base information. Typically, data base errors cause immediate termination of the execution. The system administrator should be able to assist in solving these errors which will most likely be caused by incorrect use of the system or by incorrect system installation. The ASTROS Programmer's Manual contains further information on the causes of particular data base errors.

The standard ASTROS error messages are printed by the UTMWRT utility module and represent error checks that the modules are coded to perform or errors that may cause problems in the current module's algorithm. As much as possible, these error messages are intended to be standalone in that the user should be able to interpret the message without referring to the Programmer's Manual. There are four different levels of errors that can occur in ASTROS, each labeled differently when printed:

##### (1) System Fatal Message

These messages are caused by errors or inconsistencies in the system definitions. Usually, these relate to erroneous input to the system generation utility, SYSGEN, or are a result of using an outdated system data base. You should contact your system administrator to effect a correction. Hopefully, these errors will rarely occur and should never occur in an unmodified ASTROS system.

##### (2) User Information Message

These messages are written when the system encounters data that may represent an input error or may later generate a problem but that may only be a special user input that falls outside the



expected range. Usually, these messages can be ignored. This is the least serious type of user message in ASTROS.

(3) User Warning Message

These messages are written when the system encounters data that are incorrect but which may not cause termination. In some cases, this level of error is used to signify that the system will continue to search for errors but will terminate abnormally following the search.

(4) User Fatal Message

These messages are written when the system encounters data that are in error to the extent that continuation is impossible. The system will terminate execution either immediately or after some minor clean up.

If the user is unable to decipher the error message, the following steps can help determine the source of the error:

- (1) Check the timing summary against the MAPOL sequence path to determine which module generated the error message. Also, check the SYSGEN output to determine the module that wrote the message. Note that the "message number" is included in the error message print if the message is a standard one.
- (2) Check the Programmer's Manual documentation for the relevant module to determine the error checks it performs and to get further information on the source of the error.

7.2 QUANTITIES SELECTED THROUGH SOLUTION CONTROL OPTIONS

This subsection presents a detailed description of the output quantities that can be selected through the solution control packet. These quantities fall into five categories: (1) element, (2) nodal, (3) design, (4) eigenvalues for flutter and normal modes, and (5) aeroelastic trim quantities. Each of these categories is presented in the separate subsections that follow.

The PRINT and PUNCH solution control commands are used to request the desired output quantities. These commands have three groups of options: subcase options, quantity options and form options. These options are fully described in Subsection 5.5 of this report, but one point must be stressed:

subcase options play an extremely important role in ASTROS output requests. Subcase options allow the user to identify the set of subcases to which the print selection will apply. This selection is necessary because many disciplines (MODES, for example) generate more than one subcase (e.g., eigenvector) with a single solution control directive. (The critical point is that the default selection is that there be no output.) In other words, if there are no subcases selected, ASTROS will not default to all subcases. Instead nothing will be printed.

Unlike NASTRAN, ASTROS has no options to reorder the output. The multidisciplinary nature of ASTROS completely negates the utility of NASTRAN's SORT1 and SORT2 options, and any other sort options become impossibly complex very quickly. Instead, a reasonable, fixed sort was established in which each boundary condition is treated separately and in the order given in the solution control packet. If the standard sequence is used, the response quantities will appear in the following order within each optimize or analyze boundary condition:

- (1) Steady aerodynamic trim parameters.
- (2) Flutter roots and flutter mode shape modal participation factors -- note that the mode shape is only available if flutter has occurred and if the FLUTTER discipline is within an ANALYZE boundary condition.
- (3) Applied load print requests.
- (4) The "displacement" nodal response quantities: DISPLACEMENTS, VELOCITIES, and ACCELERATIONS.
- (5) Element response quantities in the order: stress, strain, force and strain energy for each subcase. Elements are processed alphabetically within each quantity type.

In the OPTIMIZE subpacket, these data are followed by the selected design and resizing prints in the following order:

- (6) Active constraint summary (either the default abbreviated print or the full print if the DCONSTRAINT print option is selected).
- (7) The print of the global and then local design variables representing the current design -- on the final design iteration, the iteration history precedes the design variable output.

Within each response quantity's print module, the disciplines are not treated in the order given in the solution control packet; instead, they are treated, where applicable, in the following order:

- (A) STATICS
- (B) MODES
- (C) SAERO
- (D) FLUTTER
- (E) TRANSIENT
- (F) FREQUENCY
- (G) BLAST

The subcases within each discipline are treated in the order given in the solution control packet. In the case of MODES, the eigenvectors are ordered in increasing eigenvalue order. TRANSIENT, FREQUENCY, and BLAST subcases are ordered in increasing time or frequency step.

#### 7.2.1 Element Response Quantities

ASTROS has two basic forms of elements: aerodynamic elements and structural elements. An aerodynamic element is defined as a "box" of an aerodynamic macroelement, e.g., wing component or fuselage segment. The nature of the macroelement varies among both aerodynamic models and among aerodynamic components within each model. In general, however, a box is the smallest subdivision of the aerodynamic component for which data (e.g., pressures, forces, and moments) are computed. Structural elements are either metric elements, which connect structural node points (grids), scalar elements, which connect pairs of degrees of freedom or pairs of scalar points, or mass elements. Table 8 shows the list of aerodynamic and structural elements in ASTROS for which element output exists. The following subsections document the quantities that are available as output for each of these elements. The structural mass elements are not included in this table since they have no element response quantities. The NASTRAN User's Manual (Reference 2) was used as a major resource in writing this section, and the user is referred to it for additional information on the structural elements.

TABLE 8. AERODYNAMIC AND STRUCTURAL ELEMENTS IN ASTROS

<u>AERODYNAMIC</u>	<u>STRUCTURAL</u>
CAERO1 CAERO2 CAERO6 PAERO6	CBAR CELAS1, CELAS2 CIHEX1, CIHEX2, CIHEX3 CROD, CONROD CQDMEM1 CQUAD4 CSHEAR CTRMEM

Structural element output is available for all disciplines that result in a real displacement field. This includes STATICS, MODES, TRANSIENT, BLAST, and SAERO analyses. Complex displacement fields (from FLUTTER and FREQUENCY analyses) result in computation of the selected (complex) element response quantities, but their formatted print is not yet available except through executive sequence print utilities described in Subsection 7.4. For all disciplines in ASTROS, the solution control print options STRESS, STRAIN, FORCE, and ENERGY are used to select print of the structural element quantities and the PRESSURE option is used for aerodynamic element quantities. Each of these print options selects either ALL, NONE or an integer set identification number that refers to one or more ELEMIST bulk data entries specifying which elements are to have output computed and printed. Appendix D contains the complete description of the solution control print command. Each output is carefully labeled as to its boundary condition number, discipline generating the response field and load condition, mode number, time step, frequency step or flight condition represented by the output.

#### 7.2.1.1 Aerodynamic Element Output

Although there is a solution control print request for pressures at aerodynamic boxes, this feature has not been fully implemented. Instead, the aerodynamic response quantities for aerodynamic elements are selectable through modifications to the print arguments of the PFAERO and AMP modules (see Subsection 7.3).

#### 7.2.1.2 Bar Element Output

The BAR element includes extension, torsion, bending in two perpendicular planes and the associated shears. The shear center is assumed to coincide with the elastic axis. The BAR element coordinate system is shown in

**Figure 6.** The orientation of the BAR element is described in terms of two reference planes defined through the use of the orientation vector,  $\{v\}$  as shown in that figure. The positive directions for the element forces are shown in **Figure 7**. Additional information on the structural elements is contained in Section 5 of the ASTROS Theoretical Manual.

Stresses, strains, forces and strain energies are available as output for the BAR element through the STRESS, STRAIN, FORCE, and ENERGY solution control print options. The following element forces are output on request:

- (1) Bending moments at each end in both reference planes.
- (2) Shear forces in each reference plane.
- (3) Average axial force.
- (4) Torque about the bar axis.

The following element stresses and strains in the element coordinate system are output on request:

- (1) Average axial stress or strain.
- (2) Extensional stress or strain due to bending at four points on the cross-section at each end.
- (3) Maximum and minimum stress or strain at each end.
- (4) Stress margins of safety for the element in both tension and compression.

Tensile stresses and strains are given a positive value while compressive stresses and strains are given a negative value. Unlike similar output in NASTRAN, the bending contribution to the stresses are always computed at the four points on the element cross-section that were specified on the connectivity entry for the BAR element. This means that the safety margins are computed using all eight stress values even if all four stress points at each end are the same and/or coincide with the element axis. Also, margins of safety are printed even if no stress limits were given on the material entry. In these cases, a very large value for the margin of safety is used to indicate that no limits were specified. In addition, ASTROS, unlike NASTRAN, fully supports strain output for the BAR element. Strain energies may also be requested for the BAR element. The strain energy print (which is identical

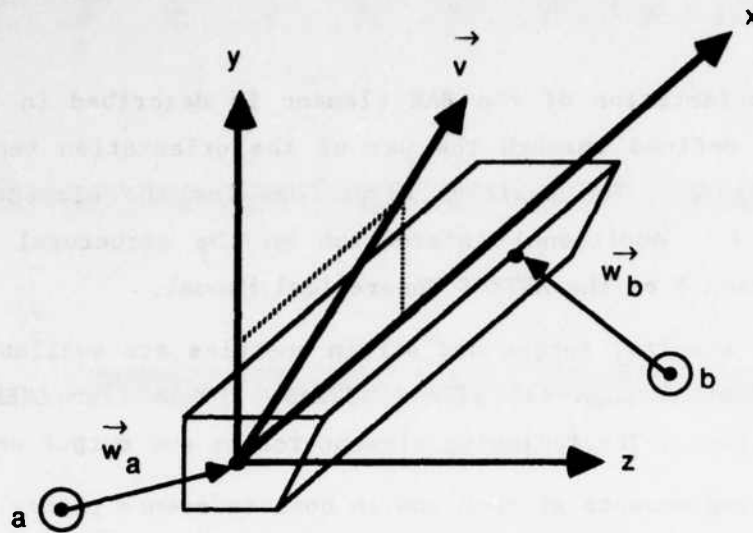


Figure 6. BAR Element Coordinate System.

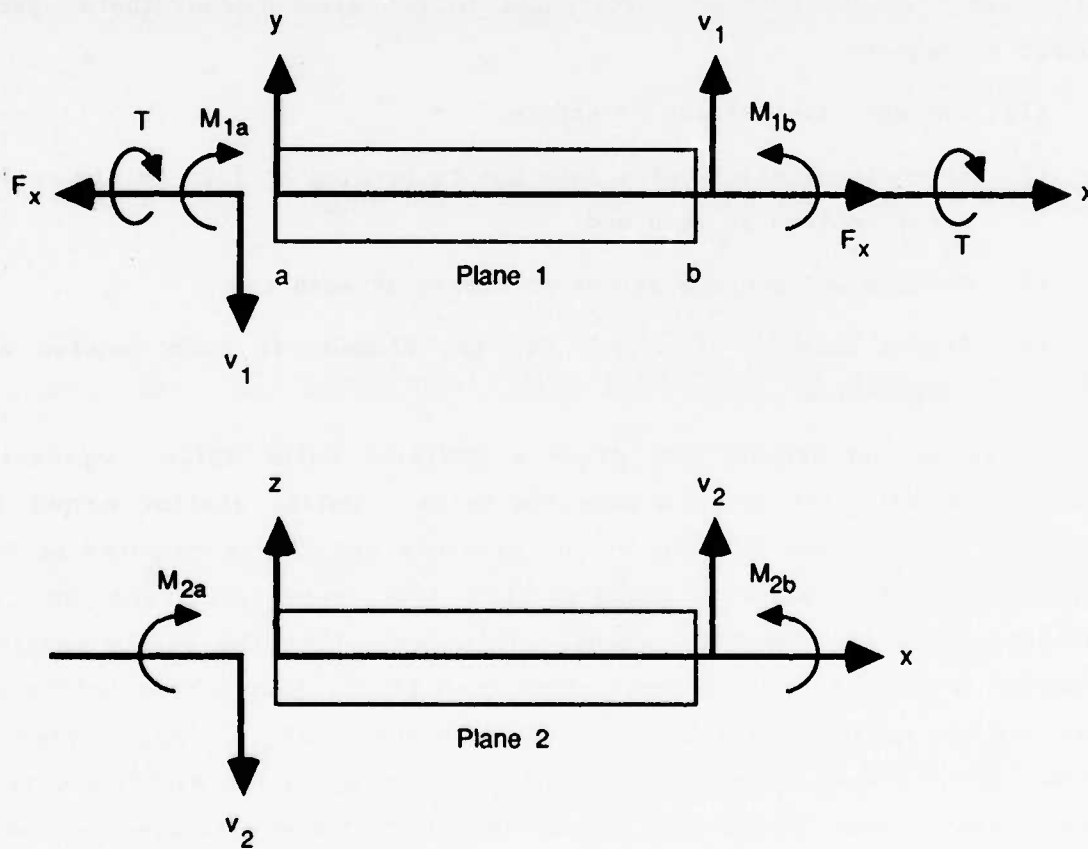


Figure 7. BAR Element Forces Sign Conventions.

for all ASTROS structural elements) is patterned after that in NASTRAN. It shows the total strain energy for the given displacement field, the strain energy in each selected element and the total strain energy for all the elements of a given type, e.g., all the BAR elements.

#### 7.2.1.3 ELAS Element Output

The ELAS element is a scalar spring element which relates the displacements at a pair of scalar points or degrees of freedom or that relates a single degree of freedom to a ground state. The element force and strain energy are directly available for the element and the user can, if desired, input a scalar quantity that relates the "stress" in the element to the displacement(s) of the connected degree(s) of freedom. On output, these values will be printed for each output request for each selected ELAS element. Strains have no meaning for the scalar spring element and any such requests will be ignored without warning. Element strain energies, however, are available for the element and are computed from the spring constant and the nodal displacement(s). The strain energy print for the ELAS is identical to that for the BAR element and includes a breakdown by element and by element type. If no scalar value is given for the element stress but the stress value is requested, a value of zero will be computed and printed for the response quantity with no warnings given.

#### 7.2.1.4 IHEX1 Element Output

The IHEX1 element is a linear isoparametric solid hexahedron element with three extensional degrees of freedom for each of its eight nodes.

Stresses, strains, and strain energies are available as output for the IHEX1 element through the STRESS, STRAIN, and ENERGY solution control print command options. Force output is not available for the IHEX1 element. On request, the following stresses and strains are output in the basic coordinate system at the center and at each corner grid point:

- (1) Normal stresses or strains in all three directions.
- (2) Shear stresses or strains in all three planes.
- (3) Principal stresses or strains in all three directions with associated direction cosines.
- (4) Mean stress or strain.
- (5) Octahedral shear stress or strain.

The stress and strain output at each of the nine points is identified by a stress or strain point ID. The stress and strain point IDs are numbered 1 through 9, with the first eight ordered as on the associated CIHEX1 input data entry, and the ninth located at the element center, as illustrated in Figure 8. All output is provided in the basic coordinate system, since there is no naturally occurring element coordinate system for the IHEX1.

Strain energy output may be requested for the IHEX1 element. The strain energy print for the IHEX1 is identical to that for the BAR element and includes a breakdown by element and by element type.

#### 7.2.1.5 IHEX2 Element Output

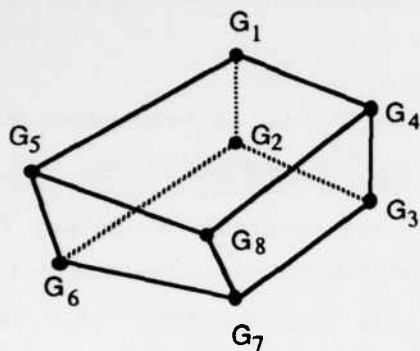
The IHEX2 element is a quadratic isoparametric solid hexahedron element with three extensional degrees of freedom for each of its 20 nodes.

Stresses, strains, and strain energies are available as output for the IHEX2 element through the STRESS, STRAIN, and ENERGY solution control print command options. Force output is not available for the IHEX2 element. On request, the following stresses and strains are output in the basic coordinate system at the 21 points located at the center, corners, and mid-edges of the element:

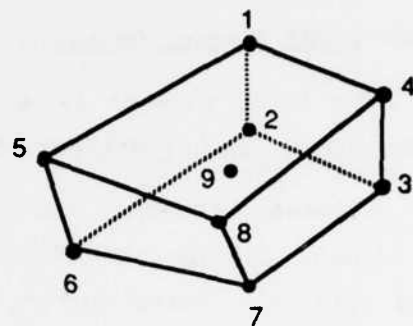
- (1) Normal stresses or strains in all three directions.
- (2) Shear stresses or strains in all three planes.
- (3) Principal stresses or strains in all three directions with associated direction cosines.
- (4) Mean stress or strain.
- (5) Octahedral shear stress or strain.

The stress and strain output at each of the 21 points is identified by a stress or strain point ID. The stress and strain point IDs are numbered 1 through 21, with the first 20 ordered as on the associated CIHEX2 input data entry, and the 21st located at the element center. Although the corner stress and strain points are located at the corner grid points of the element, the mid-edge stress and strain points may or may not be located at the mid-edge grid points, depending on the location of those grid points. The stress/strain points for the IHEX2 are illustrated in Figure 9. All output is provided in the basic coordinate system, since there is no naturally occurring element coordinate system for the IHEX2.



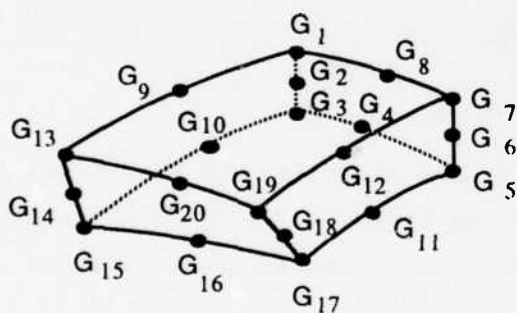


(a) ELEMENT GRID POINTS

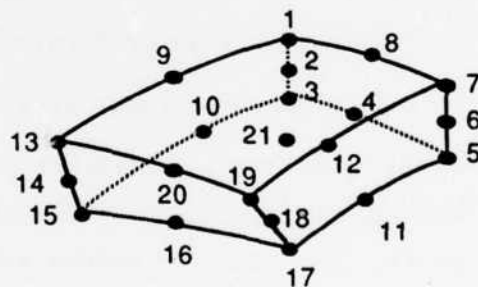


(b) ELEMENT STRESS POINTS

Figure 8. IHEX1 Element.



(a) ELEMENT GRID POINTS



(b) ELEMENT STRESS POINTS

Figure 9. IHEX2 Element.

Strain energy output may be requested for the IHEX2 element. The strain energy print for the IHEX2 is identical to that for the BAR element and includes a breakdown by element and by element type.

#### 7.2.1.6 IHEX3 Element Output

The IHEX3 element is a cubic isoparametric solid hexahedron element with three extensional degrees of freedom for each of its 32 nodes.

Stresses, strains, and strain energies are available as output for the IHEX3 element through the STRESS, STRAIN, and ENERGY solution control print command options. Force output is not available for the IHEX3 element. On request, the following stresses and strains are output in the basic coordinate system at the 21 points located at the center, corners, and mid-edges of the element:

- (1) Normal stresses or strains in all three directions.
- (2) Shear stresses or strains in all three planes.
- (3) Principal stresses or strains in all three directions with associated direction cosines.
- (4) Mean stress or strain.
- (5) Octahedral shear stress or strain.

The stress and strain output at each of the 21 points is identified by a stress or strain point ID. The stress and strain point IDs are numbered 1 through 21. The first 20 points are ordered as on the associated CIHEX3 input data entry, except that there is only one mid-edge point per edge, instead of two, and the 21st point is located at the element center. Although the corner stress and strain points are located at the corner grid points of the element, the mid-edge stress and strain points may or may not be located at a grid point, depending on the location of the mid-edge grid points. The stress/strain points for the IHEX3 are illustrated in Figure 10. All output is provided in the basic coordinate system, since there is no naturally occurring element coordinate system for the IHEX3.

Strain energy output may be requested for the IHEX3 element. The strain energy print for the IHEX3 is identical to that for the BAR element and includes a breakdown by element and by element type.

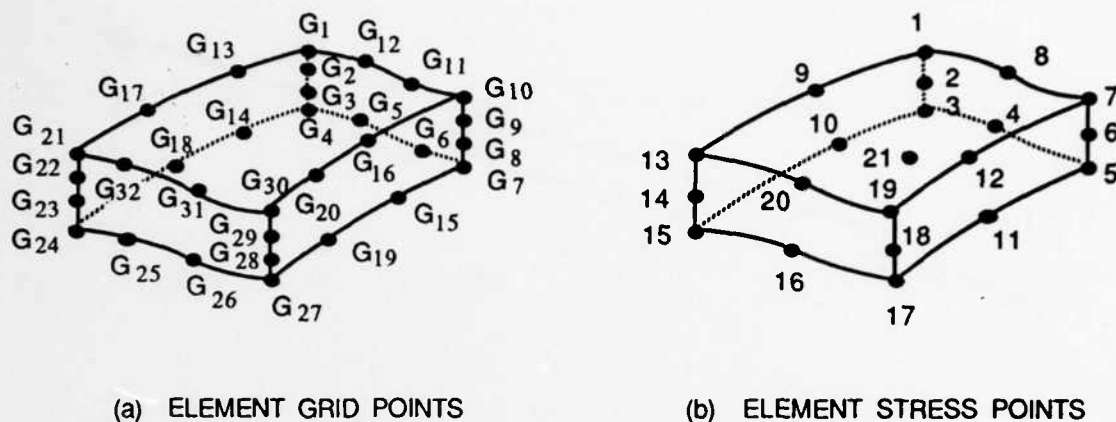


Figure 10. IHEX3 Element

#### 7.2.1.7 Rod Element Output

The ASTROS ROD element supports both extensional and rotational properties. The element coordinate system and sign conventions are shown in Figure 11. ASTROS supports stress, strain, force and strain energy output for the ROD. The forces that are computed are:

- (1) Axial force.
- (2) Torque about the element axis.

The torque and force are both computed even if the particular element does not support torsional or extensional forces, respectively. In these cases, a value of zero will be printed for the appropriate response quantity. The stresses and/or strains that are available are:

- (1) Axial stress or strain.
- (2) Torsional stress or strain.
- (3) Margin of safety for axial stress.
- (4) Margin of safety for torsional stress.

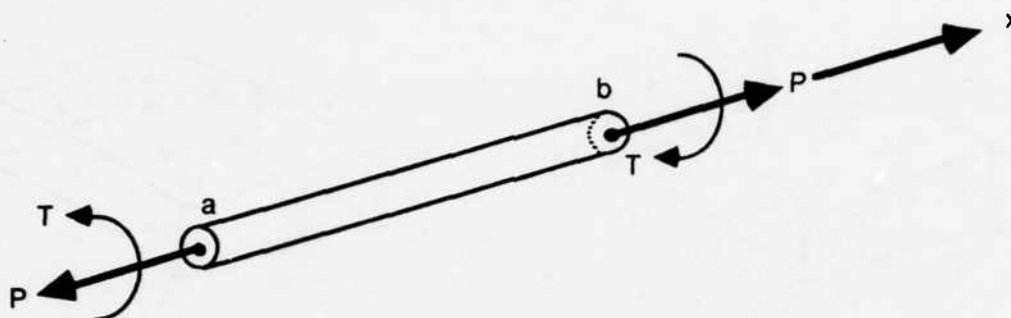


Figure 11. ROD Element Coordinate System

The margins of safety for strain are not available and the stress margins are computed even if there are no limits specified on the material property entry. In these cases, a large safety margin value is used to signify that no limits were imposed. The strain energy print for the ROD is identical to that for the BAR element and includes a breakdown by element and by element type.

#### 7.2.1.8 QDME1 Element Output

The QDME1 isoparametric, quadrilateral, membrane element supports isotropic, orthotropic and composite membrane properties. If the element is composite, the individual layers are treated as independent, stacked elements in which each "layer," as defined on the PCOMP bulk data entry, represents an element. In the case of composite elements, the layers are numbered sequentially starting with the first layer appearing on the PCOMP entry. Non-composite elements show a layer number of zero.

Stresses, strains, forces and strain energies are available for each element or layer of a composite element. Since the stresses, strains, and forces vary within a QDME1 element, the intersection point of the diagonals projected onto the mean plane of a warped element has been chosen as the point

at which the stresses, strains, forces and strain energies for the element are computed. The stresses, strains and element forces are computed in the element coordinate system. Figure 12 shows the element coordinate system and the stress computation point for the QDMEM1 element.

ASTROS computes the running loads associated with the stresses for the QDMEM1 element. These forces are:

- (1) The force components in the element coordinate system at the stress computation point.

The QDMEM1 stress and strain print includes the following:

- (1) The normal stresses or strains at the stress point in the element x- and y-directions.
- (2) The shear stress or strain on the element x face in the element y-direction.
- (3) The angle in degrees between the element x-axis and the major principal axis.
- (4) The major and minor principal (zero shear) stresses or strains.
- (5) The maximum shear stress or strain.

The strain energy print for the QDMEM1 is identical to that for the BAR element and includes a breakdown by element and by element type.

#### 7.2.1.9 QUAD4 Element Output

The QUAD4 isoparametric quadrilateral plate element includes both membrane and bending behavior. Transverse shear flexibility may be requested, as can the coupling of membrane and bending behavior. The QUAD4 element coordinate system and node numbering are shown in Figure 13. The QUAD4 element may be assigned general anisotropic or composite material properties. For designed composites, the layers are treated as stacked membrane elements similar to the QDMEM1 element. In this case, the layers are identified by number in the order specified on the PCOMP, PCOMP1 or PCOMP2 entry. For design invariant composite laminates, the output always refers to the aggregate laminate properties and refers to layer number zero. The reference plane of the QUAD4 element may be offset from the plane of the grid points and variation in the element thickness may be modeled by assigning different

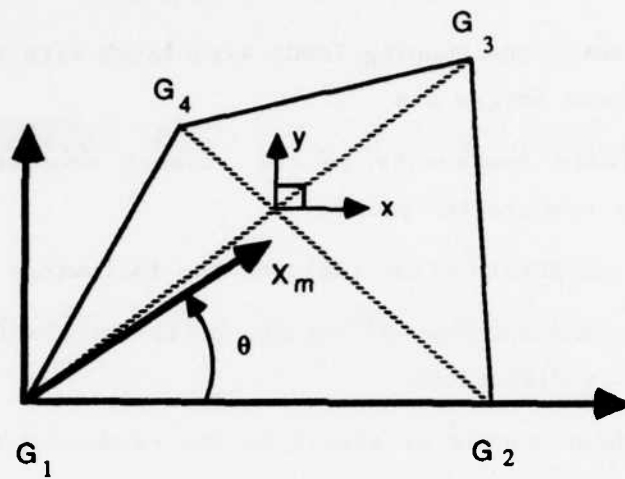


Figure 12. QDMEM1 Element Coordinate System.

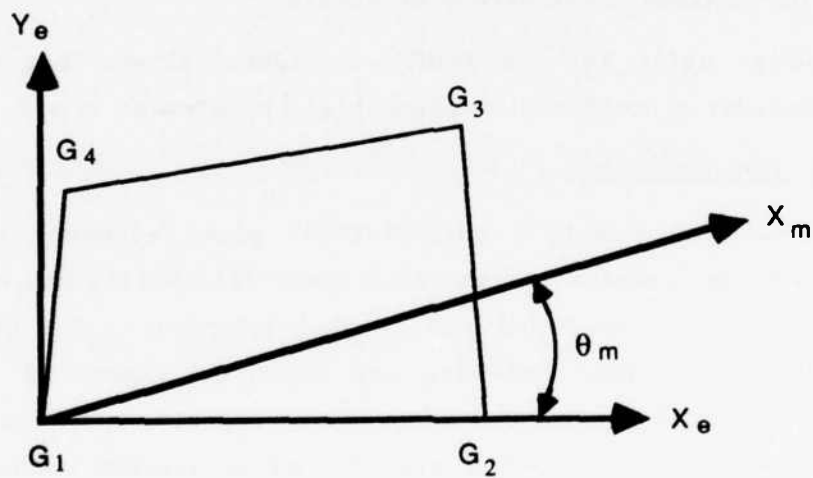


Figure 13. QUAD4 Element Coordinate System.

element thicknesses at each of the grid points. The reader is referred to Appendix A of the ASTROS Theoretical Manual for additional information on the QUAD4 element.

Stresses, strains, forces, and strain energies are available as output for the QUAD4 element through the STRESS, STRAIN, FORCE and ENERGY solution control print command options. The following element stresses and strains are output on request:

- (1) Combined extensional and bending stresses and strains computed at the element center in the element coordinate system.
- (2) Principal stresses and strains computed at the element center, including the angle between the element x-axis and the principal axis.

The following forces are output on request:

- (1) Element forces computed at the center of the element in the mean plane in the element coordinate system.

For composite materials, all output quantities are computed using the aggregate laminate properties. Hence, output of stresses or strains at the ply or laminae level is currently not an available print option for the QUAD4 element in ASTROS.

#### 7.2.1.10 Shear Panel Output

The shear panel is an element which resists the action of tangential forces applied to its edges. In ASTROS, the shear panel supports only isotropic material properties and makes use of the shear flow distribution approximation of Garvey (Reference 4) with special handling for warped, parallel edge and general trapezoidal geometries. The element coordinate system and sign convention are shown in Figure 14. The stresses, strains, forces and strain energies are available for the shear panel. The element forces that are computed include the following:

- (1) The eight forces between each pair of nodes; each force is directed along the line connecting the adjacent nodes (the element edge).
- (2) The four "kick" forces at each node, normal to the plane formed by the two adjacent element edges.
- (3) The shear flows (forces/unit length) along each edge.

When stresses or strains are requested, they are computed at the node points in skewed coordinates parallel to the adjacent edges. Both the average and maximum shear stress or strain are then printed. A safety margin based on the maximum stress value is computed for stress output. A large safety margin is printed if no limits were specified on the material property entry.

The strain energy print for the SHEAR panel is identical to that for the BAR element and includes a breakdown by element and by element type.

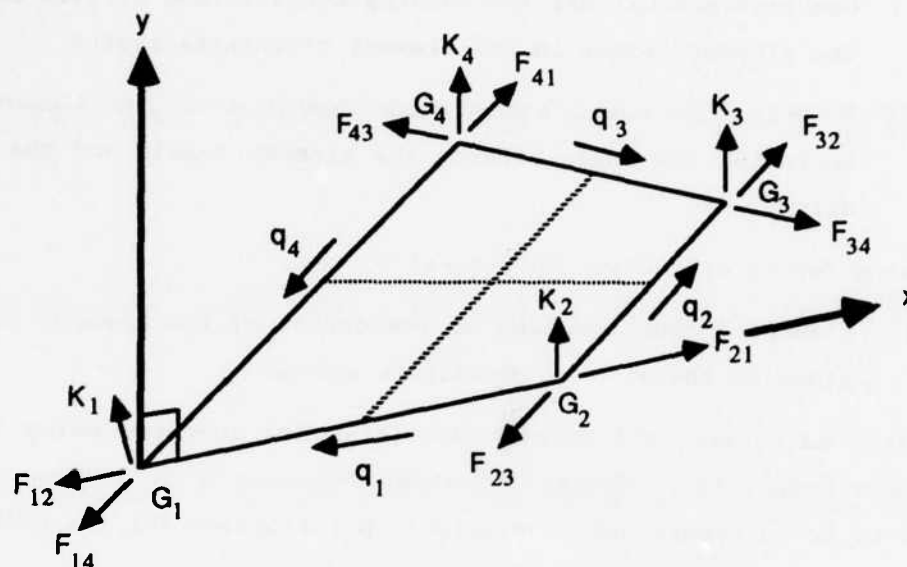


Figure 14. Shear Panel Forces and Element Coordinate System.

#### 7.2.1.11 TRMEM Element Output

The TRMEM constant strain triangular membrane element supports isotropic, orthotropic and composite membrane properties. If the element is composite, the individual layers are treated as independent, stacked elements in which each "layer," as defined on the PCOMP bulk data entry, represents an element. The layers are numbered sequentially starting with the first layer appearing on the PCOMP entry. Non-composite elements will use a layer number of zero.

Stresses, strains, forces and strain energies are available for each element or layer of a composite element. The invariant stresses, strains and forces within a TRMEM element are computed in the element coordinate system shown in Figure 15.



ASTROS computes the running loads associated with the stresses for the TRMEM element. These forces are:

- (1) The force components in the element coordinate system.

The TRMEM stress and strain print includes the following:

- (1) The normal stresses or strains in the element x- and y-directions.
- (2) The shear stress or strain on the element x face in the element y-direction.
- (3) The angle in degrees between the element x-axis and the major principal axis.
- (4) The major and minor principal (zero shear) stresses or strains.
- (5) The maximum shear stress or strain.

The strain energy print for the TRMEM is identical to that for the BAR element and includes a breakdown by element and by element type.

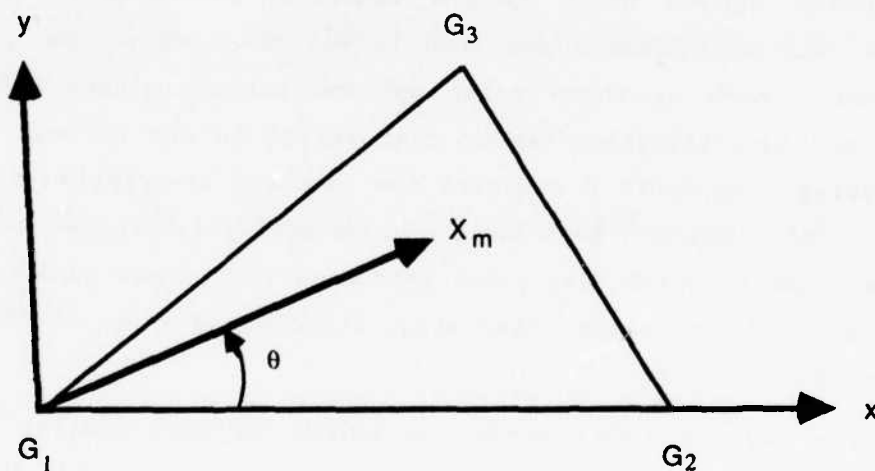


Figure 15. TRMEM Element Coordinate System.

#### 7.2.2 Nodal Response Quantities

ASTROS has two basic forms of node points: the structural node and the extra point. The structural node is defined as either a "grid" point having six degrees of freedom (three translations and three rotations) or a

"scalar" point having a single degree of freedom. These node points can be used to connect metric and scalar structural elements. The extra point is similar to the scalar point in that it has a single degree of freedom, but differs in that extra points included in the model are selected in the boundary condition rather than being implicitly included in the model. Further, they cannot be connected directly to either metric or scalar structural elements; instead, these elements are connected through terms introduced by direct matrix input or by transfer functions. Extra points are used in dynamic analyses for modeling control systems and other non-structural mechanisms in the system under analysis. These degrees of freedom do not appear in the system matrices until after the dynamic matrix assembly and do not appear in any but the dynamic response disciplines (FLUTTER, TRANSIENT, FREQUENCY and BLAST). When nodal output is requested for dynamic analyses, any extra point results may be selected using the GRIDLIST entry just as are grid and scalar point results.

Nodal output is available for all disciplines in ASTROS, although particular nodal response quantities may not be available for all disciplines. The solution control print options VELOCITY, DISPLACEMENT, GPFORCE, LOAD, SPCFORCE, and ACCELERATION are used to select print of the nodal response quantities. Each of these print options selects either ALL, NONE or an integer set identification number that refers to one or more GRIDLIST bulk data entries. Appendix D contains the complete description of the solution control print command. Each output is carefully labeled as to its boundary condition number, which discipline generated the output quantities and which load condition, mode shape, time step, frequency step or flight condition is represented by the output.

The form of nodal output in ASTROS is very similar for all nodal output quantities; therefore, only general descriptions will be given rather than individually describing each response quantity in turn. The nodal output includes the node point identification number (sorted by external identification number) and node type: (G)rid, (S)calar point or (E)xtra point. This is followed by either one or six quantities associated with the node point. The columns of the print are labeled  $T_i$  for the translations and  $R_i$  for the rotation where  $i = 1, 2$  or  $3$ . Complex nodal quantities are generated by FLUTTER and FREQUENCY disciplines and can be printed in either polar

coordinates or Cartesian coordinates through the form option on the PRINT or PUNCH command. Cartesian print is the default. Complex quantities are printed using the same columns as real nodal data but use two lines of output. The first line contains either the real part or the magnitude and the second line either the imaginary part or the phase angle in degrees.

All disciplines generate DISPLACEMENT output except some FLUTTER and SAERO analyses. Flutter mode shapes are generated only if a flutter condition occurs in the selected range of velocities and then only if the FLUTTER discipline occurs in the ANALYZE subpacket of the solution control. Steady aeroelastic analyses which do not perform a trim analysis also do not generate displacement vectors. Velocities and accelerations are only available for TRANSIENT, BLAST and FREQUENCY analyses.

The LOAD option selects output of externally applied loads at the nodal points. For STATICS, the applied mechanical, thermal and/or gravity loads are output for the selected nodes and subcases. Steady aeroelastic loads output writes, for each trim condition, those trimmed forces applied to the structure following transformation from the aerodynamic model. No loads output is available if no trim is performed in the aeroelastic analysis. TRANSIENT, FREQUENCY and BLAST disciplines compute loads at user specified time or frequency steps which may be printed. The FLUTTER discipline has no loads output.

The SPCFORCE print of single point forces of constraint has not been fully implemented and so is not available, although the solution control command is functional. The GPFORCE grid point force balance is similarly not completely installed. All the quantities needed to print a grid point force balance similar to that in MSC/NASTRAN are available on the data base but the final print modules have not been implemented. In both cases, these selections, if they appear in the solution control packet, will be ignored without warning.

### 7.2.3 Design Variables and Design Constraints

There is an important distinction between global design variables and local design variables in ASTROS. A number of linking options relating global variables to local variables are provided and are described in Section 2 of the Theoretical Manual. Briefly, the local variable is the physical element

property (e.g., thickness or cross-sectional area) that is free to change in the design process while the global variables are the actual variables that are modified by the resizing module. The resultant physical variables are then computed based on the user's linking options and the current global design variable values.

The DESIGN solution control print option allows the user to request that the global and local design variables be printed at each iteration. As discussed in Subsection 7.1, this print is the default on the last iteration. The global variable print displays the user assigned design variable identification number, the current value, the minimum and maximum values allowed for the global variable, the sensitivity of the objective function to the design variable and the linking option used to relate it to local design variables. The linking options are:

- (1) Unique Physical. The user has related the global variable to a single local variable through a DESELM entry.
- (2) Linked Physical. The user has related the global variable to some number of local variables through a combination of DESVAR/PLIST entries.
- (3) Shape Function. The user has related this and possibly other global variables to some number of local variables through a combination of DESVAR/ELIST entries.

The final item in the global design variable print is an eight-character user label identifying the design variable.

The local design variables are, of course, element dependent. Each element type that has elements connected to global design variables is printed separately. The elements are identified by element identification number and, if appropriate, the layer number. The linking option used to connect them to the global variables is also shown. This print can be very helpful in checking the correctness of the design model. Following these general data are the element dependent local variable value and the allowable range that the primary value can take. Note that the BAR element links the moments of inertia to the cross-sectional area so all three "design variables" are shown but the area is the only independent variable. The local variable print accounts for all scalar factors that might appear in the design variable

linking and therefore, indicates the true physical values represented by the current design. Finally, the two-dimensional elements include a print of the ratio of the current thickness to the minimum thickness. This additional item is included as a convenience to allow a quick computation of the number of composite plys represented by a particular design IF the user inputs the ply thickness as the minimum thickness and IF the element has composite material properties.

The solution control print option DCONSTRAINT selects that the active constraint summary print should include a table indicating which constraints are active, their current value, the constraint type and other identifying data connecting the constraints to a particular element, subcase and/or discipline. These extra data include the "type count," which is a running count (by constraint type) of all active and inactive constraints. This allows the user to identify exactly which constraint is active; e.g., the fourth flutter constraint or the 3000th von Mises stress constraint. Additionally, if the constraint is associated with a particular boundary condition, the associated boundary condition identification is shown. Similar connections are made for subcase and element dependent constraints. If the constraint is not boundary condition, subcase or element dependent, zeros or blanks will appear in the corresponding columns of the active constraint summary. The user is cautioned that the constraint ordering in the active constraint summary is not necessarily the order that constraints appear in the sensitivity matrices, the DESIGN module or other discipline dependent output.

Finally, in interpreting the constraint values, the user must be aware of some features of ASTROS design constraints. The constraints in ASTROS are all formulated such that a value greater than zero represents a violated constraint. Also, all the constraints are normalized in some manner by the design allowable. The normalization has been formulated in such a way as to provide the best behavior under the linear approximations used in the approximate optimization problem but this has the effect of obscuring the physical meaning of the constraint. The user is referred to the Theoretical Manual for the exact form of each constraint.

#### 7.2.4 Flutter/Normal Modes Response Quantities

The solution control print option ROOTS selects that the root extraction summary for flutter analyses be printed. In addition, if the flutter

analyses appear in the ANALYZE subpacket of the solution control packet, the modal participation factors of any flutter conditions will be printed. The roots are ordered such that the lowest frequency root at each velocity is associated with the lowest frequency normal mode and so on in increasing frequency order. For each normal mode, the corresponding velocity value, damping ratio, frequency and complex eigenvalue are shown. For OPTIMIZE flutter analyses, only the user's input velocities are used in the root extraction algorithm. ANALYZE flutter analyses may generate additional velocities in the process of converging to a flutter crossing.

The complex modal participation factors for each of the normal modes in the modal representation of the structure are printed if the ROOTS option is selected in ANALYZE flutter disciplines and a flutter condition is found. A flutter condition can occur for each Mach number and density ratio combination in the flutter analysis. Therefore, the flutter condition is identified by velocity, Mach number and density ratio to distinguish among multiple flutter conditions in the same analysis. Note that a zero participation factor is shown for normal modes that the user omitted from the flutter analysis.

The ROOTS print option for normal modes selects that the eigenvalue extraction table be printed. It appears immediately ahead of any eigenvectors, if any were selected. The table is patterned after that in MSC/NASTRAN and includes the eigenvalues (in sorted order), the extraction order, the cyclic and radian frequency and generalized mass and generalized stiffness for each eigenvalue extracted. The table is prefaced by data identifying the eigenvalue extraction method and some self-explanatory method dependent data.

#### 7.2.5 Aeroelastic Trim Quantities

The TRIM solution control print option selects that the aeroelastic trim parameters and stability coefficients be printed. There are three types of aeroelastic trim analyses in ASTROS: (1) the zero degree of freedom trim (or no trim); (2) the single degree of freedom (lift) longitudinal trim; and (3) the two degrees of freedom (lift and pitching moment) longitudinal trim. The zero degree of freedom "trim" may be either longitudinal or lateral. Each TRIM print is labeled with the Mach number, dynamic pressure, reference grid point, and the appropriate normalization parameters. These parameters are the

reference area and chord length for longitudinal coefficients and reference area and span for lateral coefficients.

The longitudinal trim print includes, in the most general case, the lift and pitching moment stability coefficients for:

- $C_{L_o}, C_{M_o}$  - Thickness and camber effects
- $C_{L_\alpha}, C_{M_\alpha}$  - Angle of attack ( $\alpha$ ) in both radians and degrees
- $C_{L_\delta}, C_{M_\delta}$  - Elevator deflection ( $\delta$ ) in both radians and degrees
- $C_{L_q}, C_{M_q}$  - Pitch rate ( $q$ ) in both radians and degrees.

These nondimensional factors are implicitly defined in the following equations:

$$\text{Lift} = \bar{q} S \left[ C_{L_o} + C_{L_\alpha} \alpha + C_{L_\delta} \delta + C_{L_q} \frac{qc}{2V} \right]$$

$$\text{Pitching Moment} = \bar{q} S c \left[ C_{M_o} + C_{M_\alpha} \alpha + C_{M_\delta} \delta + C_{M_q} \frac{qc}{2V} \right]$$

where,

- $\bar{q}$  - Dynamic Pressure
- $S$  - Reference Area
- $c$  - Reference Chord
- $V$  - Reference Velocity

These definitions are the standard forms used in aircraft stability and control (see Reference 5).

Each of these pairs of quantities (lift and pitching moment coefficients) is shown in three forms:

- (1) The stability derivative for the rigid aerodynamic model as computed directly from the forces acting on the aerodynamic boxes (termed DIRECT in the output).

- (2) The stability derivative for the rigid aerodynamic model as computed from the forces transformed to the structural degrees of freedom (termed SPLINED in the output).
- (3) The flexible derivative which includes corrections for the flexibility and inertia relief effects.

If the first two forms do not agree closely (within 1 to 2 percent), the spline transformation may be incorrect and the first step to correct this would be to check the SPLINE1 and ATTACH data entries. The second and third forms are used in the lift effectiveness calculations.

Finally, the trim parameters that were computed for the current flight condition are shown. In general, these are the angle of attack in degrees and the elevator deflection angle in degrees. For single degree of freedom trim (i.e., lift equals weight), the elevator stability coefficients and trim parameter will not appear in the print.

The lateral trim print is only applicable to zero degree of freedom trim analyses in which the SUPORT degree of freedom is an axial roll degree of freedom. In this case, the rolling moment stability coefficients associated with the following quantities are shown:

$C_{L_{\delta_a}}$  - Rolling moment due to aileron deflection ( $\delta$ ), in both radians and degrees.

$C_{L_p}$  - Rolling moment due to roll rate ( $p$ ), in both radians and degrees.

These nondimensional factors are implicitly defined in the following equation:

$$\text{Roll} = qSb \left[ C_{L_{\delta_a}} \delta_a + C_{L_p} \frac{pb}{2V} \right]$$

where,

$b$  - reference semispan

These quantities are shown in three forms:



- (1) The stability derivative for the rigid aerodynamic model as computed directly from the forces acting on the aerodynamic boxes (termed DIRECT in the output).
- (2) The stability derivative for the rigid aerodynamic model as computed from the forces transformed to the structural degrees of freedom (termed SPLINED in the output).
- (3) The flexible derivative which includes corrections for the flexibility and inertia relief effects.

As in the longitudinal case, discrepancies between the data for the first two forms may indicate an error in the spline transformations. The aileron effectiveness calculation is performed using the third form. There are no trim parameters printed with the lateral stability coefficients.

### 7.3 EXECUTIVE SEQUENCE SELECTABLE QUANTITIES

The MAPOL compiler supports a feature to declare that certain arguments to modules (external procedures or functions) be optional. These have been used in several ASTROS modules for debug print requests. A number of modules, therefore, have "hidden" arguments which do not appear in the standard sequence, but which may be included if a nonzero or nonblank value (the defaults for optional MAPOL arguments are zero or blank, as appropriate) is desired. This subsection documents the optional arguments to ASTROS modules that relate to modifying the level or form of output from the module. These options are also presented in the appropriate Programmer's Manual sections.

#### 7.3.1 Bulk Data Echo Options

The input file processor module, IFP, has the two optional arguments, SORT and ECHO, in its calling sequence:

```
CALL IFP (GSIZE, sort, echo);
```

Both SORT and ECHO are integer arguments that control the form and destination for the echo of the bulk data packet. The following options are available:

<u>SORT</u>	<u>ECHO</u>	<u>ACTION</u>
0	0	Sorted echo to the output file.
0	1	No echo to either output or punch files.
0	2	Sorted echo to punch file.
0	>2	Sorted echo to both output and punch files.
>0	0	Unsorted echo to the output file.
>0	1	No echo to either output or punch files.
>0	2	Unsorted echo to punch file.
>0	>2	Unsorted echo to both output and punch files.

The default is that sorted echo is written to the output file and nothing is written to the punch file.

### 7.3.2 Intermediate Steady Aerodynamic Matrix Output

The preface aerodynamic module, PFAERO, has a single optional integer print debug as the last argument of it calling sequence:

```
CALL PFAERO (Gsize,      [AICMAT(MINDEX)],      [AAICMAT(MINDEX)],
              [AIRFRM(MINDEX)], MINDEX, NAERO, [GTKG], [GSTKG], [UGTKG],
              [AJJTL], [D1JK], [D2JK], [SKJ], print);
```

The PRINT option will generate output from the USSAERO submodule of the preface aerodynamics module PFAERO. There are five print levels available:

<u>PRINT</u>	<u>ACTION</u>
0	No printed output is generated.
1	Prints steady aerodynamic model geometry and a few miscellaneous debugs.
2	Prints the above and stability coefficient data.
3	Prints the above and pressure data from the USOLVE submodule.
4	Prints the above and voluminous data from the calculation of velocity components and intermediate matrices from the USSAERO submodule.

The user is cautioned that a print level of 4 generates a large amount of data. Most of these prints are vestigial prints from the USSAERO code that was adapted for use in the ASTROS system. In cases where the output data are not self-evident, the user is referred to the USSAERO documentation (Reference 6).

### 7.3.3 Intermediate Unsteady Aerodynamic Matrix Output

The secondary unsteady aerodynamic preface module, AMP, has an optional integer print argument as the last argument in its calling sequence and an optional matrix argument as the second to last:

```
CALL AMP ([AJJTL], [D1JK], [D2JK], [SKJ], [QKKL], QKJL, [QJJL],  
         [ajjdc], print);
```

The PRINT option controls the output of several intermediate matrices or of individual matrices from the matrix lists QKKL, QKJL and QJJL that are formed in AMP for FLUTTER, GUST and BLAST analyses, respectively. The user is referred to the Programmer's Manual for complete documentation of these data base entities. The following matrices are output:

<u>PRINT</u>	<u>ACTION</u>
0	No output.
1	(1) The SKJ matrix. If there is only one aerodynamic group, the following are also printed:  (2) If FLUTTER entries are in the bulk data packet, the matrix, [X], representing the solution to the equation: $[AJJ] * [X] = ([D1JK] + (ik)[D2JK])$  (3) If GUST entries are in the bulk data packet, the matrix [QKJ] from the corresponding matrix list.  (4) If BLAST entries are in the bulk data packet, the matrix [QJJ] from the corresponding matrix list.
>1	All of the above and:  (1) If FLUTTER entries are in the bulk data packet, the matrices [D1JK] and [D2JK].  (2) The matrix [AJJT] after extraction from the corresponding matrix list.

The optional matrix [AJJDC] is used to store the intermediate matrix [X] described in the options shown above. If [AJJDC] is blank, a scratch data base entity is used to store [X]. In either case, [X] may be printed through the PRINT option. Only the last [X] matrix calculated will be returned to the executive sequence in [AJJDC] for use in additional processing.

#### 7.3.4 Flutter Root Iteration Output

The flutter analysis module, FLUTTRAN, has an optional trailing integer print argument to generate additional information on the flutter eigenvalue extraction:

```
CALL FLUTTRAN (BC, [QHHL(BC)], LAMBDA, HSIZE, [MHH(BC)],  
              [KHHF(BC)], print);
```

The PRINT option in this case pertains to prints that give information on the iterative solution of the flutter matrices. The PRINT option has the following meaning:

<u>PRINT</u>	<u>ACTION</u>
0	No output.
1	Print the number of iterations required to find each flutter root.
>1	Print the above plus information on each of the estimated roots for each iteration. This voluminous information may sometimes be of use, when the flutter solution goes astray, in determining if a modified set of velocities would give improved results.

#### 7.3.5 Stress Constraint Computation Output

The stress/strain constraint evaluation module, SCEVAL, has an optional integer print parameter as its final argument:

```
CALL SCEVAL ( BC, [GLBSIG], [UG(BC)], TRMTYP, print );
```

The PRINT argument, if nonzero, will generate a listing, by element type, of all the constrained elements, the current value of their stress components and the resultant constraint value for each design load condition. Also included in the print is the running "type count" for stress and strain constraints that appear in the Active Constraint Summary print described in Subsection 7.2.3. This allows the user to identify exactly which elements and subcases are associated with each particular stress or strain constraint. This print is a remnant from the ASTROS development when the element stresses were not available, but it may still be useful in checking out the constraint modeling for large problems.

### 7.3.6 Intermediate Optimization Output

The DESIGN module for resizing via mathematical programming methods has a final optional integer print argument that selects a print of intermediate data:

```
CALL DESIGN (CONVERGE, MOVLIM, CNVRGLIM, CTL, CTLMIN, OPSTRAT,  
            NUMOPTBC, [AMAT], print);
```

The PRINT argument is passed directly to the MICRODOT optimization package which makes the following intermediate quantities available:

<u>PRINT</u>	<u>ACTION</u>
0	No output is generated.
1	Initial design information and final results.
2	The above and function values at each iteration.
3	The above and internal MicroDOT parameters.
4	The above and search directions.
5	The above and gradient information.
6	The above and scaling information.
7	The above and one-dimensional search information.

These PRINT options allow the user to view the detailed calculations used in the solution of the approximate constrained optimization problem that ASTROS generates at each iteration. The user is cautioned that the data printed from the DESIGN module are not necessarily ordered in the same manner as in other design prints and not identified by user supplied design variable identification numbers.

### 7.4 EXECUTIVE SEQUENCE OUTPUT UTILITIES

In recognition of the inability to provide for the print of all useful response quantities, utilities have been included in the set of MAPOL modules to augment the solution control print options. These utilities may be placed in any MAPOL program where the user desires to see additional information. These utilities print the data contained in general data base entities. The formats of these prints are more general and therefore, less well identified than the special print options described in the preceding subsections. The

generality of these utilities, however, is felt to be a vital addition to the output features of the ASTROS procedure in that almost any data on the user's data base files can be written to the output file. These utilities provide a primitive link between ASTROS and external post-processing systems.

#### 7.4.1 Structural Set Definition Print Utility. USETPRT

The USETPRT utility has been provided to print, for each boundary condition in the solution control packet, the structural set definition table stored in the ASTROS data base entity, USET. This utility exactly mimics the capabilities provided by the NASTRAN PARAM/USETPRT option. The USETPRT module has the following calling sequence:

```
CALL USETPRT ( BC );
```

For the selected boundary condition, BC, each degree of freedom in the structural model is listed in a table which shows the structural sets to which the degree of freedom belongs. The reader is referred to Section 4 of the Theoretical Manual for more information on the structural set definitions in ASTROS.

#### 7.4.2 Special Matrix Print Utility. UGPRT

The print utility UGPRT has been provided in order to view particular matrices whose rows correspond to the structural degrees of freedom. In general, these matrices are very large and virtually impossible to interpret without some additional formatting beyond that which is available for more general matrix prints. Therefore, for the supported matrix entities, the matrix columns are printed in a form similar to that used for the nodal response quantities as presented in Subsection 7.2.2. The UGPRT utility has the following calling sequence:

```
CALL UGPRT ( BC, [mat1], [mat2], ... [mat10] );
```

where up to ten matrix arguments may be supplied. The BC integer argument identifies the associated boundary condition so that the utility can make use of the USET entity in formatting the output. The following entities are supported:

[DKUG], [DMUG], [DPVJ], [DUG], [DPGV], [DUGV]

and

[DPTHVI], [DPGRVI]

and

[PG]

and

[DFDU]

The utility keys off the entity names, so the above names must be used, although the matrices can be subscripted if the user wishes. The reader is referred to the Programmer's Manual for additional information on the data contained in these entities.

#### 7.4.3 General Matrix Print Utility. UTMPT

The matrix print utility UTMPT has been written such that any data base matrix entity can be printed to the output file. The calling sequence for UTMPT is:

```
CALL UTMPT ( method, [mat1], [mat2], ... [mat10] );
```

where up to ten matrices can be printed in a single call. The optional integer METHOD argument selects from among two formats that are available. If METHOD is zero or absent, the entire matrix column, starting with the first nonzero term and ending with the last nonzero term, will be printed, including all intermediate zeros. If METHOD is nonzero, only the nonzero terms of each column will be printed.

#### 7.4.4 General Relation Print Utility. UTRPT

The print utility UTRPT has been written such that any data base relational entity can be printed to the output file. The calling sequence for UTRPT is:

```
CALL UTRPT ( rel1, rel2, ... rel10 );
```

where up to ten relations can be printed in a single call. Relational entities are tables in which the columns are called attributes. The UTRPT utility attempts to print the relation in a format in which each column represents one attribute and each row represents a single entry in the rela-

tion. The utility is not very sophisticated, however, and relations having more attributes than can fit in the width of a page (128 characters or approximately 12 attributes) will have the trailing attributes ignored. Also, string attributes are only printed if they are eight characters long. Despite its limitations, the UTRPRT utility can be very useful in viewing ASTROS data base relations.

#### 7.4.5 General Unstructured Print Utility, UTUPRT

The print utility UTUPRT has been written such that any data base unstructured entity can be printed to the output file. The calling sequence for UTUPRT is:

```
CALL UTUPRT ( UNSTRUCT, type );
```

Unlike other data base entities, there is no information in an unstructured entity to identify what type of data is stored in its records. The user, therefore, must supply the TYPE argument to select the proper format to use in the print. The following TYPE's are available:

TYPE < 0 prints only the record length (in single precision words) of each record in the entity

TYPE = 0 prints each record using an integer format

TYPE = 1 prints each record using a real single precision format

TYPE = 2 prints each record using a double precision format.

For TYPE values greater than or equal to zero, each record will be printed, in its entirety, in the selected format. If, as is typical, the record contains mixed data, e.g., both integer and real data, the user can make multiple calls to UTUPRT to view first the integer format and then the real format. No error will occur, but the real data will give spurious looking integer prints and vice versa.

#### 7.5 ASTROS OUTPUT LIMITATIONS

Post-processing can often generate heated debate among users regarding the lack of foresight of the system developers in failing to provide a simple method to obtain a particular response quantity. It is appropriate to address this issue in this section. A code like ASTROS (and NASTRAN) is so general that to make all possible response quantities available would take decades of effort, if it could be done at all. In fact, a mature finite element code



like NASTRAN is continuously adding new output capabilities as the user community dictates. Naturally, a new code like ASTROS cannot (and should not) attempt to address all these features. Instead, the ASTROS designers considered it important to focus on design optimization and provide a large, but finite, number of options for post-processing outside this area. Among those options, however, is the capacity to obtain printed output for all the data storage forms on the ASTROS data base. In addition, the output processing software was carefully designed such that additional quantities are relatively simple to install in the system. Thus, while ASTROS cannot yet provide for the output of all useful response quantities, the capability to provide them at some future date has been addressed and, of more immediate importance, a set of utilities has been written to provide generic output of the data base entities. These general utilities, in combination with the MAPOL language, make many more quantities available to the user than are shown in the preceding subsections. The user is cautioned, therefore, to keep in mind the flexibility of the ASTROS system and not to become overly constrained by the limitations in the output processor. With only a little knowledge of the engineering software and of the MAPOL language, the user will find that a large number of additional quantities can be computed and printed.

## REFERENCES

1. Herendeen, D. L., Hoesly, R. L. and Johnson, E. H., "Automated Strength-Aeroelastic Design of Aerospace Structures," AFWAL-TR-85-3025, September 1985.
2. The NASTRAN User's Manual (Level 17.5), National Aeronautics and Space Administration, NASA SP-222(05), December 1978.
3. Schaeffer, H.G., MSC/NASTRAN Primer: Static and Normal Modes Analysis, Schaeffer Analysis, Inc., Mount Vernon, New Hampshire, 1982.
4. Garvey, S. J., "The Quadrilateral Shear Panel," Aircraft Engineering, May 1951, p. 134.
5. Etkin, B., Dynamics of Flight, John Wiley and Sons, Inc., New York, May 1967.
6. Woodward, F. S., "USSAERO Computer Program Development, Versions B and C," NASA CR 3227, 1980.

## APPENDIX A

### ASSIGN DATABASE DESCRIPTIONS

This appendix contains the descriptions of the machine and installation dependent parameters on the ASSIGN DATABASE entry for four machines on which ASTROS is currently functional. The parameters that are available at each site are listed and details of their use are presented. The user is cautioned that these are site dependent parameters which may be different for each installation even if the host system is the same. This documentation is provided both as an example to the system programmer and because the features that have been made available on these machines are very likely to exist on most machines that may be used. The user is referred to his/her ASTROS system manager for the particulars of the interface between the local host system and ASTROS.

The host systems that are documented are:

- (1) The MicroVAX II and VAX 8600 using VMS
- (2) The IBM 370 mainframe
- (3) The Perkin Elmer 30xx

Note that the two VMS systems are identical.

## A.1 VMS IMPLEMENTATION

The ASSIGN DATABASE entry supplies the ASTROS system with the root name of the data base files <dbroot>, the status of those files, and a set of user parameters. The root name is limited to eight characters in length, the status is one of NEW, OLD or TEMP and the set of user parameters can be any of the following keyword commands:

VOL = <vol>	Names the device and/or default directory on which the data base files reside.
DBLKSIZ = <n>	CADDB data files block size in words.
IBLKSIZ = <n>	CADDB index file block size in words.
DELETE	Denoting that the run-time data base files are to be deleted after the execution.
KEEP	Denoting that the run-time data base files are to be kept after the execution.

### DCL Requirements:

Under MicroVMS and VMS, the data base files can be explicitly named in the FORTRAN open statement or an equivalence between a logical name and the true file name can be established using the DCL ASSIGN command:

```
$ ASSIGN physical-name logical-name
```

Both options are used in the VMS version of ASTROS. If no VOL parameter is specified, the filenames dbrootIDX and dbrootD01 are used in the OPEN statement. These are typically ASSIGNED to the physical names in the DCL COM file. When a VOL parameter is specified, however, a file name with the structure of a physical file name is used:

```
voldbroot.IDX
```

```
voldbroot.D01
```

As an example:

```
ASSIGN DATABASE MULT SHAZAM NEW
```

would use MULTIDX and MULTD01 as file names in FORTRAN OPEN statements whereas:

```
ASSIGN DATABASE MULT SHAZAM NEW VOL = $DISK1:[SCR]
```

would use \$DISK1:[SCR]MULT.IDX and \$DISK1:[SCR]MULT.DO1 as filenames in the OPEN statements.

#### TEMP Data Base Example:

When the status is "TEMP," a temporary data base is created. Internally, the file is not named in the OPEN statement and has a status of "SCRATCH". Generally, this means that the files will be created in the user's default directory. The only legal optional parameters are DBLKSIZE and IBLKSIZE. In the example:

```
ASSIGN DATABASE TEST PASS TEMP
```

The data base files will be created using the default block sizes and deleted upon termination of the execution.

#### NEW Data Base Example:

When the status is "NEW," a new data base is created. The filename will be formed from the root name and, if present, from the VOL option as discussed. Any of the user parameters may be used. As an example:

```
ASSIGN DATABASE TEST PASS NEW
```

will create either the files TESTIDX and TESTDO1 or, if the DCL contains the proper assignments:

```
$ ASSIGN SYS$SCRATCH:TEST.IDX TESTIDX
```

```
$ ASSIGN SYS$SCRATCH:TEST.DO1 TESTDO1
```

the files named in the DCL ASSIGN statements.

The DELETE option will cause the files to be deleted via the DISP = 'DELETE' option on the OPEN and CLOSE statements. Otherwise the files will be kept.

#### OLD Data Base Example:

When the status is "OLD," an old data base is used. The physical files that make up the data base must exist. Only the VOL parameter is legal and the data base files will be kept after the execution has terminated. In the example:

```
ASSIGN DATABASE TEST PASS OLD VOL = $DISK1:[DB]
```

the preexisting data base files

\$DISK1:[DB]TEST.IDX

\$DISK1:[DB]TEST.D01

will be used in the ASTROS execution.

#### A.2 IBM 370 IMPLEMENTATION

The ASSIGN DATABASE entry supplies the ASTROS system with the root name of the data base files, <dbroot>, the status of those files, and a set of user parameters. The root name is limited to five characters in length, the status is one of NEW, OLD or TEMP and the set of user parameters can be any of the following keyword commands:

DBLKSIZ = <n>            CADDB data file block size in words.

IBLKSIZ = <n>            CADDB index file block size in words.

#### NEW Data Base Example:

When the status is "NEW," a new data base is created. The NEW refers to the fact that the data base itself is to be initialized. The physical files that make up the data base may exist or be created with the proper DD card. Other legal parameters are DBLKSIZ and IBLKSIZ. Here is an example:

ASSIGN DATABASE RICH RICHPASS NEW

JCL should be as follows:

//RICHIDX DD DSN=RICHDB.INDEX.SPACE=(TRK,10),DISP=(,CATLG)

//RICHDO1 DD DSN=RICHDB.DATA,SPACE=(TRK,50),DISP=(,CATLG)

In this example a data base with files RICHDB.INDEX and RICHDB.DATA will be created.

#### JCL Requirements:

On the IBM computer the user must supply the appropriate JCL cards for each data base file. The DD names for these cards are built from the root file name, <dbroot>, as follows:

dbrootIDX - index file  
dbrootD01 - 1st data file  
dbrootD02 - 2nd data file

.  
.  
.

The only required DD parameters for new files are SPACE and DISP. The user can control the number of data files by including the desired number of DD cards.

#### TEMP Data Base Example:

When the status is "TEMP," a temporary data base is created. Internal to the data base, there is no difference between a 'TEMP' and a 'NEW' data base on the IBM. The only difference is in the JCL where the user supplies a temporary file instead of a permanent one. The only legal optional parameters are DBLKSIZE and IBLKSIZE. Here is an example:

```
ASSIGN DATABASE RICH RICHPASS TEMP DBLKSIZE=2048 IBLKSIZE=256
```

JCL should be as follows:

```
//RICHIDX DD DSN=&INDEX,SPACE=(TRK,10)  
//RICHDO1 DD DSN=&DATA,,SPACE=(TRK,50)
```

#### OLD Data Base Example:

When the status is "OLD," an old data base is used. The physical files that make up the data base must exist. No other parameters are legal. Here is an example:

```
ASSIGN DATABASE RICH RICHPASS OLD
```

JCL should be as follows:

```
//RICHIDX DD DSN-RICHDB.INDEX,DISP=OLD  
//RICHDO1 DD DSN-RICHDB.DATA,DISP=OLD
```

### A.3 PERKIN ELMER IMPLEMENTATION

The ASSIGN DATABASE entry supplies the ASTROS system with the root name of the data base files, <dbroot>, the status of those files, and a set of user parameters. The root name is limited to eight characters in length, the

status is one of NEW, OLD or TEMP and the set of user parameters can be any of the following keyword commands:

VOL - <vol>	Names the disk volume on which the data base files reside.
DBLKSIZ - <n>	CADDB data file block size in words.
IBLKSIZ - <n>	CADDB index file block size in words.
DELETE	Denoting that the run-time data base files are to be deleted after the execution.
KEEP	Denoting that the run-time data base files are to be retained after the execution.

#### TEMP Data Base Example:

When the status is "TEMP," a temporary data base is created. Perkin Elmer temporary files are used so no data are kept after the run. The only legal optional parameters are DBLKSIZ and IBLKSIZ.

```
ASSIGN DATABASE RICHDB RICHPASS TEMP DBLKSIZ=2048 IBLKSIZ=256
```

#### NEW Data Base Example:

When the status is "NEW," a new data base is created. The NEW refers to the fact that the data base itself is to be initialized. The physical files that make up the data base may exist or be created. If the files exist, the DELETE or KEEP parameter is used to determine whether the existing files are to be deleted and reallocated or kept. Two files are used to store the data base. The names of these files are as follows:

```
vol:dbroot.IDX - index file
```

```
vol:dbroot.D01 - data file 1
```

The VOL parameter and the root data base name are used to form the names. Other legal parameters are DBLKSIZ and IBLKSIZ.

```
ASSIGN DATABASE RICHDB RICHPASS NEW DELETE, VOL=SYSR
```

In this example, a data base with files SYSR:RICHDB.IDX and SYSR:RICHDB.D01 will be created. If existing files are found they will be deleted and reallocated.



OLD Data Base Example:

When the status is "OLD," an old data base is used. The physical files that make up the data base must exist. Two files are used to store the data base. The names of these files are as follows:

vol:dbroot.IDX - index file

vol:dbroot.D01 - data file 1

The VOL parameters and the root data base name are used to form the names. No other parameters are legal. Here is an example:

ASSIGN DATABASE RICHDB RICHPASS OLD VOL-SYSR

In this example a data base with files SYSR:RICHDB.IDX and SYSR:RICHDB.D01 will be used.

APPENDIX B  
MAPOL PROGRAMMER'S MANUAL

This appendix contains the programmer's manual for the ASTROS executive language, MAPOL. This manual is a standalone document which presents the syntax and features of the MAPOL language. It contains the general information needed to make syntactically correct modifications to the ASTROS standard MAPOL sequence and to write independent MAPOL programs to direct the ASTROS system. All variable types, statement forms, input/output features and intrinsic functions are presented. Because the manual is standalone, the section numbering contained in this Appendix is inconsistent with the remainder of the report and a separate table of contents, figures and tables have been provided.

## TABLE OF CONTENTS

<u>SECTION</u>	<u>PAGE</u>
I	INTRODUCTION AND USER OPTIONS. . . . .
	143
1.1	User Options. . . . .
	143
1.2	MAPOL Program Form. . . . .
	144
1.3	The Standard ASTROS Solution. . . . .
	145
1.4	Modifying the Standard Solution . . . . .
	145
1.5	Creating MAPOL Programs . . . . .
	146
1.6	Summary . . . . .
	146
II	DATA TYPES AND USER OPTIONS. . . . .
	147
2.1	Introduction. . . . .
	147
2.2	Definitions and Notation. . . . .
	147
2.3	Simple Data Types . . . . .
	148
2.3.1	Data Type INTEGER. . . . .
	148
2.3.2	Data Type REAL . . . . .
	148
2.3.3	Data Type COMPLEX. . . . .
	149
2.3.4	Data Type LOGICAL. . . . .
	149
2.3.5	Data Type LABEL. . . . .
	149
2.4	Complex Data Types. . . . .
	149
2.4.1	Data Type MATRIX and IMATRIX . . . . .
	150
2.4.2	Data Type RELATION . . . . .
	150
2.4.3	Data Type UNSTRUCT and IUNSTRUCT . . . . .
	152
2.4.4	Data Base Entity Declaration Requirements. . . . .
	153
III	EXPRESSIONS AND ASSIGNMENTS. . . . .
	155
3.1	Introduction. . . . .
	155
3.2	Arithmetic Expressions. . . . .
	155
3.2.1	Arithmetic Operators . . . . .
	155
3.2.2	Arithmetic Operands. . . . .
	155
3.2.3	Evaluation of Arithmetic Expressions . . . . .
	156
3.2.4	The Use of Parentheses . . . . .
	156
3.2.5	Type and Value of Arithmetic Expressions . . . . .
	157
3.3	Logical Expressions . . . . .
	158
3.3.1	Logical Operators. . . . .
	158
3.3.2	Logical Operands . . . . .
	158
3.3.3	Evaluation of Logical Expressions. . . . .
	159
3.4	Relational Expressions. . . . .
	159
3.4.1	Relational Operators . . . . .
	160
3.4.2	Relational Operands. . . . .
	160
3.4.3	Evaluation of Relational Expressions . . . . .
	160

# TABLE OF CONTENTS

<u>SECTION</u>		<u>PAGE</u>
	3.5 Matrix Expressions. . . . .	160
	3.5.1 Matrix Operators . . . . .	160
	3.5.2 Matrix Operands and Expressions. . . . .	162
	3.6 Assignment Statements . . . . .	162
IV	CONTROL STATEMENTS . . . . .	163
	4.1 Introduction. . . . .	163
	4.2 The Unconditional GOTO Statement. . . . .	163
	4.3 Iteration . . . . .	163
	4.3.1 The FOR...DO Loop. . . . .	164
	4.3.2 The WHILE...DO Loop. . . . .	165
	4.4 The IF Statement. . . . .	166
	4.4.1 The Logical IF . . . . .	166
	4.4.2 The Block IF . . . . .	166
	4.4.3 The IF...THEN...ELSE . . . . .	167
	4.4.4 Nested IF Statements . . . . .	167
	4.5 The END and ENDP Statements . . . . .	168
V	INPUT/OUTPUT STATEMENTS. . . . .	169
	5.1 Introduction. . . . .	169
	5.2 The PRINT Statement . . . . .	169
VI	PROCEDURES AND FUNCTIONS . . . . .	171
	6.1 Introduction. . . . .	171
	6.2 Program Units and Scope of Variables. . . . .	171
	6.3 Defining a Procedure. . . . .	172
	6.4 Invoking a Procedure. . . . .	173
	6.5 Function Procedures . . . . .	173
	6.6 Intrinsic Function Procedures and Intrinsic Procedures. . . . .	174
	6.7 Intrinsic Mathematical Functions. . . . .	174
	6.8 Intrinsic Relational Procedures . . . . .	175
	6.9 General Intrinsic Procedures. . . . .	177

# LIST OF TABLES

<u>TABLE</u>		<u>PAGE</u>
B-1	Command Options. . . . .	144
B-2	Summary of User Options. . . . .	146
B-3	Arithmetic Operations in MAPOL . . . . .	155
B-4	Rules for X OP Y, where OP is +, -, * or / . . . . .	157
B-5	Rules for Exponentiation, X ** Y . . . . .	158
B-6	Allowable Logical Operators. . . . .	158
B-7	Evaluation of Logical Expressions. . . . .	159
B-8	Relational Operators in MAPOL. . . . .	160
B-9	Matrix Operations in MAPOL . . . . .	161
B-10	Assignment Rules in MAPOL. . . . .	162
B-11	Intrinsic Mathematical Functions in MAPOL. . . . .	175
B-12	Intrinsic Relational Procedures in MAPOL. . . . .	175

## LIST OF ILLUSTRATIONS

<u>FIGURE</u>		<u>PAGE</u>
B-1	Schematic Representation of Relational Data. . . . .	151
B-2	A MAPOL Program to Compute Cube Roots. . . . .	165
B-3	A MAPOL Procedure to Determine the Square Root of a Number . .	173
B-4	A MAPOL Program Using Intrinsic Relational Procedures. . . . .	177

## SECTION I

### INTRODUCTION AND USER OPTIONS

The Matrix Analysis Problem Oriented Language (MAPOL) is a high level computer language that has been designed to support the large-scale matrix operations typically encountered in engineering analysis. Its conceptual roots may be traced to the Direct Matrix Abstraction Program (DMAP) capability found in the NASTRAN<sup>®</sup> system developed by NASA in the late 1960's. The DMAP "language," used to create NASTRAN's solution algorithms is very crude; however, it has been a prime factor in extending the life cycle of the system. It has done this by providing a simple method of installing new code and functional capabilities into the system. It also affords the user an opportunity to interact with the software.

MAPOL has been selected to provide the same advantages to the ASTROS system. Additionally, it extends the primitive DMAP design by assimilating the many advances that have been made in computer science over the last two decades.

MAPOL is a structured procedural language that directly supports high order matrix operations, the manipulation of data base entities and complex data types. The syntax of the language is similar to PASCAL, and it should be easily learned by anyone familiar with FORTRAN or PASCAL.

This manual describes in detail all of the features of the MAPOL language and gives numerous examples of their use.

#### 1.1 USER OPTIONS

In this section, the different kinds of MAPOL programs and their uses are discussed. MAPOL is the control language of the ASTROS system and, as such, the multidisciplinary solution algorithm is simply a MAPOL program that is embedded in the system. The user is free to modify this standard algorithm and can also create individual MAPOL programs or specialized procedures.

## 1.2 MAPOL PROGRAM FORM

If an ASTROS analysis is not using the standard solution, then a MAPOL program is required as the first part of the input data stream. The MAPOL data packet must be formed as shown:

```
MAPOL [ <option-list> ] [;]  
...  
...  
...  
END;
```

In this manual, all capitalized and underscored words (e.g., MAPOL) must appear exactly as they are written. A symbol enclosed in angle brackets (e.g., <option-list>) represents one or more choices to be made. If the symbol is enclosed in square brackets (e.g., [<id>]), the choice is optional.

The MAPOL command, which must be the first statement in the program, selects compiler options. These options are shown in Table B-1 where the "D" indicates the default option.

TABLE B-1. MAPOL COMMAND OPTIONS

NAME	OPTION
<u>LIST</u> (D) <u>NOLIST</u>	Lists the MAPOL source program
<u>GO</u> (D) <u>NOGO</u>	Selects, or deselects, execution after program compilation

As an example, the statement:

```
MAPOL NOLIST;
```

will cause the MAPOL program to be compiled and executed with no listings produced, while the statement:

```
MAPOL NOGO;
```

will cause the program to be compiled and a listing of the source code produced. After compilation, ASTROS will terminate without executing the program.



### 1.3 THE STANDARD ASTROS SOLUTION

As mentioned earlier, the ASTROS multidisciplinary solution algorithm is a MAPOL program. The code resides on the ASTROS system data base. It is retrieved and used whenever a MAPOL program is not found in the input data stream. A listing of the standard solution algorithm is included in the output from the ASTROS system generation program, SYSGEN.

### 1.4 MODIFYING THE STANDARD SOLUTION

In some cases, the user may wish to modify the standard ASTROS solution in order to, for example, perform some auxiliary computations not currently available or to execute only a portion of the solution. Special MAPOL editing commands allow for these modifications:

DELETE a [ , b ]

REPLACE a [ , b ]

INSERT a

DELETE is used to remove one or more statements starting with line "a" and, optionally, ending with line "b" inclusively. REPLACE performs a deletion of the specified line, or lines, and replaces them with any following MAPOL statements. The INSERT command allows any number of MAPOL statements to be inserted after line "a".

For example:

EDIT;

INSERT 1

\$ MY MODIFICATION \$

REPLACE 20,23

A := 2 \* B;

DELETE 101,237

Note that rather than entering the MAPOL command, the special EDIT declaration is used.

In the example, a comment is added at the beginning of the algorithm to document the modification. Several lines (20-23) are replaced by a new computational expression, and a larger block of lines (101-237) are removed from the program.

### 1.5 CREATING MAPOL PROGRAMS

If the standard executive sequence is not selected, the MAPOL compiler assumes that a new program is being created. This new program may perform any operations that use any of the matrix and data base utilities available in the ASTROS system. All of these are described in subsequent chapters of this manual.

### 1.6 SUMMARY

Table B-2 summarizes the MAPOL statements that have been described in this section, along with their uses.

TABLE B-2. SUMMARY OF USER OPTIONS

STATEMENT	FUNCTION
<u>MAPOL</u> [<option-list>]	Begin a MAPOL program and to select its name and compiler options
<u>END</u>	Terminate the MAPOL program
<u>EDIT</u>	Modify the standard solution
<u>DELETE</u> a[,b]	Remove lines when editing
<u>REPLACE</u> a[,b]	Remove and insert lines when editing
<u>INSERT</u> a	Insert lines when editing

## SECTION II

### DATA TYPES AND DECLARATIONS

#### 2.1 INTRODUCTION

This section describes the data types that are available in the MAPOL language. It discusses their specification during programming and how they are represented in the ASTROS machine.

#### 2.2 DEFINITIONS AND NOTATION

All programming languages are composed of two kinds of symbols. The first kind of symbol is an explicit part of the language. In MAPOL, such symbols include special characters such as:

= - \* := ;

and "reserved words" such as:

REAL RELATION IF ELSE WHILE

In this manual, reserved words are indicated by capitalized and underscored names.

The second kind of symbol is an identifier, or variable name, which may be chosen by the programmer. Identifiers are composed of letters and digits, but the first character must always be a letter. This and other definitions in this manual are shown as:

$\langle \text{ident} \rangle ::= \langle \text{letter} \rangle \mid \langle \text{ident} \rangle \langle \text{letter} \rangle \mid \langle \text{ident} \rangle \langle \text{digit} \rangle$

The vertical line "|" is read as "or". This definition clearly specifies all possible legal identifiers, because no matter how many times the rules:

$\langle \text{ident} \rangle ::= \langle \text{ident} \rangle \langle \text{letter} \rangle$

or

$\langle \text{letter} \rangle ::= \langle \text{ident} \rangle \langle \text{digit} \rangle$

are used, the user must finally use the rule:

$\langle \text{ident} \rangle ::= \langle \text{letter} \rangle$

This final rule ensures that the identifier begins with a  $\langle \text{letter} \rangle$ . In MAPOL  $\langle \text{letter} \rangle$  refers to any of the upper case letters from "A" to "Z", and  $\langle \text{digit} \rangle$  to the integers from "0" to "9".

Note that although this open-ended definition of an identifier (such a definition is called "recursive") implies that arbitrarily long names may be used, the MAPOL compiler has an "implementation limit" of eight characters for a variable name.

### 2.3 SIMPLE DATA TYPES

The MAPOL language supports five simple data types:

- o INTEGER
- o REAL
- o COMPLEX
- o LOGICAL
- o LABEL

MAPOL is a strongly typed language, and as such, all variables must be declared at the beginning of a program unit. This is done with one or more declaration statements. The syntax of a declaration statement is defined by the rules shown below:

```
<decl>      := <type> <var-list>
<type>      := REAL | INTEGER | COMPLEX | LOGICAL | LABEL
<var-list>  := <var> | <var>, <var-list>
```

Each simple variable, with the exception of LABEL, may be an array with one or two subscripts. This is defined by:

```
<var>       := | <ident> (<sub1>) | <ident> (<sub1> , <sub2>)
<sub1>      := INTEGER
<sub2>      := INTEGER
```

#### 2.3.1 Data Type INTEGER

INTEGER's are whole numbers such as 157, 83 or 22. An INTEGER may also have a sign associated with it such as -47 or +1024. The range of integers depends upon the ASTROS host computer.

#### 2.3.2. Data Type REAL

REAL data represents floating point numbers. Such numbers include 1.75, 0.00025, 1.78E-6 and -3.00271E+36.

REAL numbers are represented in a manner determined by the "machine precision" of the host computer automatically. MAPOL, therefore, does not distinguish between the REAL and DOUBLE PRECISION types such as is found in FORTRAN.

#### 2.3.3 Data Type COMPLEX

COMPLEX numbers are those which may be represented in the form:

$$a + bi$$

Because some host computers automatically handle COMPLEX data while others do not, MAPOL and ASTROS handle such data in a manner totally independent of the host computer.

In the ASTROS machine, both "a" and "b" are represented as a pair of machine precision floating point numbers. Most available mathematical functions operate on COMPLEX data.

#### 2.3.4 Data Type LOGICAL

LOGICAL variables have a value of "true" or "false". The ASTROS machine represents "true" by the integer value "1" and "false" by the integer value "0" regardless of the host computer convention.

#### 2.3.5 Data Type LABEL

LABEL's are used to define statement locations within a MAPOL program. Typically, they are only used with the GOTO statement (see Section 4).

### 2.4 COMPLEX DATA TYPES

To best support comprehensive engineering analysis capabilities, MAPOL supports five complex, or high level, data types:

- o MATRIX, IMATRIX
- o RELATION
- o UNSTRUCT, IUNSTRUCT

All of these types represent data base entities. Matrices and unstructured entities may be handled only in their entirety in MAPOL. Relations may be accessed on an entry-by-entry and attribute-by-attribute basis. Use of the

IMATRIX and IUNSTRUCT, ("I" for indexed) data types allows for more efficient retrieval of data that are accessed in a random order.

#### 2.4.1 Data Types MATRIX and IMATRIX

Matrix data base entities are declared in a slightly different manner from the remaining data types. The rules for their declaration are:

```
<decl>      := MATRIX <mat-list>
<mat-list>  := <mat-list> , <mat-var> | <mat-var>
<mat-var>   := [<ident>] | [<ident> (sub)]
<sub>       := INTEGER
```

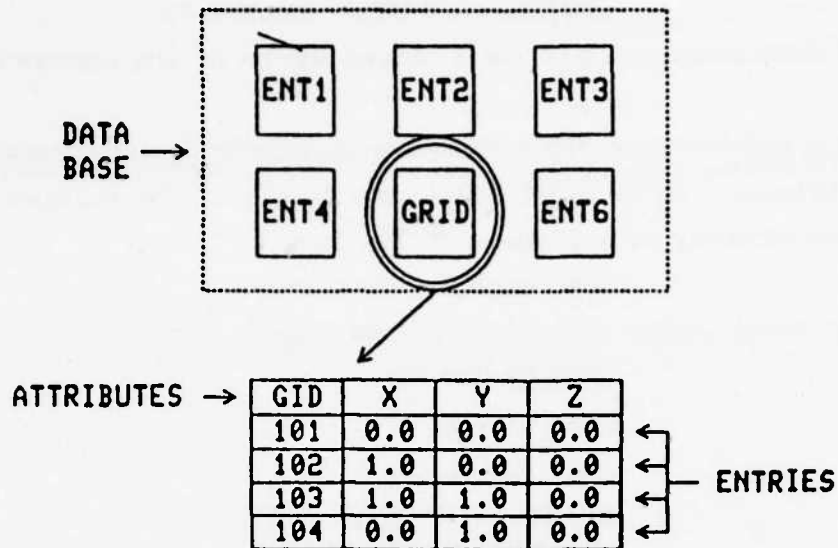
Note that the matrix <ident> is enclosed in square brackets (i.e., []) for clarity and ease-of-reading of MAPOL programs. Matrix expressions, then, look as they do written in standard mathematical notation. Matrix variables may also be subscripted to allow multiple entities to be referenced using the same <ident>. This feature is used in ASTROS to allow data from multiple boundary conditions to be saved for subsequent sensitivity evaluation. There is an implementation limit of two subscripts which may take on any integer value from 1 to 1000. When subscripted matrix entities are used, the executive system generates a CADDB entity name and relates that name to the subscript value. The MAPOL programmer is therefore cautioned that, unlike other high order variables, subscripted matrix entities do not have a corresponding CADDB entity of the same name.

#### 2.4.2 Data Type RELATION

The most complex and powerful MAPOL data type is the RELATION. Briefly, a relation can be thought of simply as a table. The rows of the table are called "entries" and the columns "attributes." The CADDB is a collection of such relations (Figure B-1). In the figure, a single relation, called GRID, has been highlighted. The GRID relation has four attributes: an identification number (GID) and three spatial coordinates (X,Y and Z). The schemas of relations that reside on CADDB are fixed.

All of the relations that are generated by the ASTROS modules that appear in the MAPOL program must be declared. The rules for these

## CADDDB - RELATIONAL ENTITIES



**Figure B-1. Schematic Representation Of Relational Data**

declarations are:

```

<decl>      := RELATION <rel-list>
<rel-list> := <rel-list> , <rel-var> | <rel-var>
<rel-var>  := <ident>
    
```

If the user wishes to use the individual attributes of a relation, or to define a new relation, the PROJECT declaration is used:

```

<decl>      := PROJECT <rel-var> USING <att-list>
<rel-var>  := <ident>
<att-list> := <att-list> , <attname> | <attname>
<attname> := <ident>
    
```

The names of each of the attributes, <attname>, must match those defined in the CADDDB schema if the relation already exists. Otherwise, they are used to define the schema for the new relation.

As an example, the GRID relation of Figure B-1 would be:

```
INTEGER GID;  
REAL X,Y,Z;  
PROJECT GRID USING GID,X,Y,Z;
```

Note that each attribute must be declared and be of the appropriate type.

Once a relation and its projection have been declared, specific entries may be retrieved. After a retrieval, any or all of the relation's attributes may be used directly by variables of the form:

<relname> @ <attname>

This illustrated in the following program segment:

```
INTEGER GID,ID;  
REAL X,Y,Z;  
REAL C1,C2,C3;  
PROJECT GRID USING CID,X,Y,Z;  
...  
...  
ID := GRID@GID;  
C1 := GRID@X;  
C2 := GRID@Y;  
C3 := GRID@Z;  
...
```

The value of an attribute within a relation may be modified if an assignment is made and then the entry is written onto CADDB (refer to Section 6.8).

#### 2.4.3 Data Types UNSTRUCT and IUNSTRUCT

The simplest CADDB data structure is called an UNSTRUCTured entity. The form and content of such an entity is the responsibility of the ASTROS programmer. The only use of the UNSTRUCT entity declarations is to type those which are used for inter-module communications: UNSTRUCT entities, which may not be subscripted, are declared with:

```
<decl>      := UNSTRUCT <un-list>  
<un-list>   := <un-list> , <un-var> | <un-var>  
<un-var>    := [<ident>]
```



#### 2.4.4 Data Base Entity Declaration Requirements

All of the ASTROS data base entities may be divided into three classes: (1) MAPOL entities, (2) HIDDEN entities, and (3) TEMPORARY entities. MAPOL entities are those that are used and appear in the MAPOL program such as matrices or relations used in calculations and any entity appearing as an argument in a functional module call. HIDDEN entities represent data that are used by a functional module but whose contents are generated from required physical data. As an example, the GRID Bulk Data are stored in a relation called 'GRID'. Many modules might wish to access this GRID data. Requiring the GRID relation to appear in the calling list of each such module is more disruptive than it is beneficial. As a result, GRID might never explicitly appear in the MAPOL program. It must, however, be declared so that the CADDB will be properly initialized. The last entity type, the TEMPORARY entity, is used mostly as a "scratch" area for intra-module use. As such it is created and deleted by the module needing it.

In summary, all of the MAPOL and HIDDEN entities must be declared in the MAPOL program. TEMPORARY entities are not declared.

## SECTION III

### EXPRESSIONS AND ASSIGNMENTS

#### 3.1 INTRODUCTION

In this section, the relationships between the various data types are described. Of particular importance is the manner in which data are combined by arithmetic expressions and how values are assigned to the ASTROS machine memory.

#### 3.2 ARITHMETIC EXPRESSIONS

Arithmetic expressions are formulae for computing numeric values. An arithmetic expression consists of either a single operand or two or more operands separated by arithmetic operators.

##### 3.2.1 Arithmetic Operators

MAPOL supports five arithmetic operators as shown in Table B-3.

TABLE B-3. ARITHMETIC OPERATORS IN MAPOL

OPERATOR	DESCRIPTION
+	Addition when connecting two operands. Unary plus when preceding an operand.
-	Subtraction when connecting two operands. Negation when preceding an operand.
*	Multiplication.
/	Division.
**	Exponentiation.

Successive operands must be separated by operators, and two operators may not be used in succession.

##### 3.2.2 Arithmetic Operands

Arithmetic operands may be constants, symbolic names of constants, variables (including relational attributes), array elements, or function references. Operands may also be arithmetic expressions and arithmetic expressions enclosed in parentheses. The data type of an arithmetic operand

may be INTEGER, REAL or COMPLEX. In some cases, it may also be MATRIX. It may never be LOGICAL, LABEL, RELATION (without an attribute specification), or UNSTRUCT.

### 3.2.3 Evaluation of Arithmetic Expressions

Expressions are evaluated from left to right according to the following hierarchy of operations:

FIRST	FUNCTION EVALUATION
SECOND	**
THIRD	* AND /
FOURTH	+ AND -

This hierarchy is used to determine which of two sequential operations is to be performed first. If two sequential operations are of equal rank, the operations are performed from left to right except when the operators are both exponentiation operators, in which case the operations are performed right to left. If two sequential operations are of unequal rank, the higher ranking operation is performed first.

When a unary minus or plus appears in an arithmetic expression, it follows the same hierarchy as a minus or plus used for subtraction or addition. For example:

$R = -S**T$  is evaluated as  $R = -(S**T)$

$R = -S/T$  is evaluated as  $R = -(S/T)$

$R = -S+T$  is evaluated as  $R = (-S)+T$

The division of operands in an expression may result in a truncated value for integer operands or a fractional value for non-integer operands. Therefore, parentheses should be used when a specific order of evaluation other than left to right is desired for the operands. For example, the expression  $8*7/4$  has a resultant value of 14; the expression  $8*(7/4)$  has a resultant value of 8. Similarly, the expression  $3.0/2.0*6.0$  has a resultant value of 9.0; the resultant value of the expression  $3.0/(2.0*6.0)$  is 0.25.

### 3.2.4 The Use of Parentheses

Parentheses may be used in arithmetic expressions to specify the order

of operation. This allows an evaluation that is different from the standard hierarchy. Whenever parentheses are used, the enclosed expression is evaluated prior to its use. When such expressions are nested, the innermost expressions are evaluated first.

The expression

$$X := A - \text{SQRT}(B) / (C-D) * E^{**}2 (F-G);$$

is therefore evaluated in the following order:

```

SQRT(B)      -> TEMP1
(C-D)        -> TEMP2
TEMP1/TEMP2   -> TEMP3
E ** 2       -> TEMP4
TEMP3*TEMP4   -> TEMP5
(F-G)        -> TEMP6
TEMP5*TEMP6   -> TEMP7
A - TEMP7     -> X

```

### 3.2.5 Type and Value of Arithmetic Expressions

Type conversions are performed when mixed expressions are evaluated. The final value of an arithmetic expression may depend upon this type conversion. Tables B-4 and B-5 show the conversions that occur when two operands are combined with an arithmetic operator:

TABLE B-4. RULES FOR X OP Y, WHERE OP IS +, -, \* OR /

X \ Y			
	INTEGER	REAL	COMPLEX
INTEGER	INTEGER	REAL	COMPLEX
REAL	REAL	REAL	COMPLEX
COMPLEX	COMPLEX	COMPLEX	COMPLEX

**TABLE B-5. RULES FOR EXPONENTIATION,  $X ** Y$**

X \ Y	INTEGER	REAL	COMPLEX
INTEGER	INTEGER	REAL	ILLEGAL
REAL	REAL	REAL	ILLEGAL
COMPLEX	COMPLEX	ILLEGAL	ILLEGAL

Special rules apply to operations when one or more of the operands is of the type MATRIX. These rules are discussed in Section 3.5.

### 3.3 LOGICAL EXPRESSIONS

A logical expression produces a logical data type result with a value of TRUE or FALSE.

#### 3.3.1 Logical Operators

Table B-6 lists the logical operators that may be used in logical expressions.

**TABLE B-6. ALLOWABLE LOGICAL OPERATORS**

OPERATOR	DESCRIPTION
<u>NOT</u>	Negation ( Unary )
<u>AND</u>	Conjunction
<u>OR</u>	Disjunction
<u>XOR</u>	Equivalence

Logical operators must be separated by logical operands except for the following two cases:

AND NOT

OR NOT

#### 3.3.2 Logical Operands

Any of the following operands may be used in logical expressions:

- o LOGICAL CONSTANTS
- o LOGICAL VARIABLES
- o LOGICAL ARRAY ELEMENTS

- o LOGICAL FUNCTION REFERENCE
- o LOGICAL EXPRESSION
- o RELATIONAL EXPRESSION

Both logical and relational expressions may be enclosed in parentheses.

### 3.3.3 Evaluation of Logical Expressions

Logical expressions are evaluated based on "truth tables" shown in Table B-7. L1 and L2 are logical variables, T and F signify TRUE and FALSE:

TABLE B-7. EVALUATION OF LOGICAL EXPRESSIONS

VARIABLES		RESULT			
L1	L2	<u>NOT</u> L1	L1 <u>OR</u> L2	L1 <u>AND</u> L2	L1 <u>XOR</u> L2
T	T	F	T	T	F
T	F	F	T	F	T
F	T	T	T	F	T
F	F	T	F	F	F

Logical operators have a hierarchy similar to the arithmetic operations:

FIRST	LOGICAL FUNCTION
SECOND	<u>NOT</u>
THIRD	<u>AND</u>
FOURTH	<u>OR</u>
FIFTH	<u>XOR</u>

Any operation in a logical expression may be enclosed in parentheses; the parenthetical expression is evaluated, and the resulting value is used as an operand. Thus, parentheses may be used to alter the order in which operations are to be performed. When parenthetical expressions are nested, evaluation begins with the innermost set of parentheses and proceeds to the outermost set.

### 3.4 RELATIONAL EXPRESSIONS

A relational expression uses relational operators to compare two arithmetic expressions. A relational expression produces a logical data type

with a value of TRUE or FALSE. Thus, a relational expression may be an operand in a logical expression.

#### 3.4.1 Relational Operators

Table B-8 summarizes the relational operators available in MAPOL.

TABLE B-8. RELATIONAL OPERATORS IN MAPOL

OPERATOR	DESCRIPTION
=	EQUAL TO
<>	NOT EQUAL TO
>	GREATER THAN
>=	GREATER THAN OR EQUAL TO
<	LESS THAN
<=	LESS THAN OR EQUAL TO

#### 3.4.2 Relational Operands

Relational operands must be of an arithmetic type integer or real. A complex operand is permitted only when the relational operator is = or < >.

#### 3.4.3 Evaluation of Relational Expressions

In a relational expression involving the comparison of arithmetic operands, each of the arithmetic operands is evaluated prior to testing the relation. When the data type of two arithmetic operands differs, one operand is converted to the type of the other before the comparison is made. (See Section 3.2.5 for data type conversions.) The numeric values of the arithmetic operands are compared as specified by the relational operator, and the resulting value is either TRUE or FALSE.

### 3.5 MATRIX EXPRESSIONS

Matrix expressions are those which combine two or more matrices to yield a matrix result.

#### 3.5.1 Matrix Operators

MAPOL allows four computational matrix operators as shown in Table B-9.

TABLE B-9. MATRIX OPERATORS IN MAPOL

OPERATOR	DESCRIPTION
$A + B$	$a_{ij} + b_{ij}$
$A - B$	$a_{ij} - b_{ij}$
$A * B$	$\sum_k a_{ik} b_{kj}$
$-A$	$-a_{ij}$

All matrices must be conformable in order to perform these operations. In the case of addition and subtraction, this means that the number of rows and columns in A and B must be the same. In the case of multiplications, the number of columns of the premultiplier must equal the number of rows in the postmultiplier.

Matrix equations are written with the square brackets just as they are when declared. Examples of these equations are:

$[A] := [B] * [C];$

$[X] := [Q(I)] * [Z];$

$[P(2)] := [R] - [S(2 * K + 1)];$

Matrix operands may also be grouped to direct the order of operation. Instead of the parentheses used in scalar expressions, the square brackets are again used as shown below:

$[A] := [[B] + [C] * [D]] + [E];$

$[A(I)] := [[B] * [[C] + [D] * [E]]] * [F];$

All matrix algebra is optimized to provide the most effective use of computer resources. Matrices may also be multiplied by scalars or scalar expressions which may be INTEGER, REAL or COMPLEX. These operations are written in the natural way; e.g.:

$[A] := (x) [B];$

$[Q(2)] := (R + S * T) [C] + [D];$

All scalar multipliers must be placed on the left as shown.

In addition to the matrix algebra operations of Table B-9, MAPOL allows for matrix transpose and inverse using the syntax of the following example:



```

[A] := TRANS([B]) * [C];
[X] := INV([KGG])*[PG];
[U] := TRANS([A])*INV([A]*TRANS([A]))*[B];

```

Note that these operations are functions and, as such, the arguments are enclosed in parenthesis.

### 3.5.2 Matrix Operands and Expressions

Only matrix operands may be used in matrix expressions with the exception noted in Section 3.5.1. The matrix expressions are evaluated with the same hierarchy as that of arithmetic types.

### 3.6 ASSIGNMENT STATEMENTS

Assignment statements are used to compute and assign values to variables and array elements. The syntax of a MAPOL assignment is:

`<var> := <expr>`

The type of the expression `<expr>` is converted to the type of the variable `<var>` based on the rules of Table B-10 where the following definitions are used.

- VAL(x) - value of x
- FIX(x) - convert x to an integer value
- FLOAT(x) - convert x to a floating point value
- REAL(c) - convert to the real part of a complex number

**TABLE B-10. ASSIGNMENT RULES IN MAPOL**

TYPE OF <VAR>	TYPE OF <EXPR>	ASSIGNMENT RULE
INTEGER	INTEGER	VAL(<EXPR>) INTO <VAR>
	REAL	FLOAT(VAL (<EXPR>)) INTO <VAR>
	COMPLEX	FIX(REAL(VAL(<EXPR>))) INTO <VAR>
REAL	INTEGER	FLOAT(VAL(<EXPR>)) INTO <VAR>
	REAL	VAL(<EXPR>) INTO <VAR>
	COMPLEX	REAL(VAL(<EXPR>)) INTO <VAR>
COMPLEX	INTEGER	FLOAT(VAL(<EXPR>) INTO REAL <VAR>
	REAL	VAL(<EXPR>) INTO REAL(<VAR>)
	COMPLEX	VAL(<EXPR>) INTO <VAR>

## SECTION IV

### CONTROL STATEMENTS

#### 4.1 INTRODUCTION

Control statements are statements used to alter and control the normally sequential execution of MAPOL instructions. There are five MAPOL control statements:

- o GOTO
- o FOR ... DO
- o WHILE ... DO
- o IF ... THEN ... ELSE
- o END , ENDP

#### 4.2 THE UNCONDITIONAL GOTO STATEMENT

The GOTO statement causes the MAPOL program to jump unconditionally to the specified statement label. This label must exist in the same program unit (see Section 6) as the GOTO statement. The label identifier must also have been declared in the program unit's specification statements. The general syntax is:

GOTO <label>;

where <label> is any legal identifier that has been declared.

The label is denoted by <label>: followed by any valid MAPOL statement. As an example;

```
IF A < B GOTO SKIP;
:
:
SKIP: C:: B -A;
```

Note that the label must be followed by a colon.

#### 4.3 ITERATION

It is often necessary to execute a group of statements repeatedly. Generally, although the statements themselves remain the same, the data on which they operate changes. This iteration or "looping" must terminate after a finite number of iterations; therefore, a decision must be made to determine whether to continue or terminate the loop. MAPOL supports two

iteration forms: FOR...DO and WHILE...DO. Each is described in this section.

#### 4.3.1 The FOR...DO Loop

It is often necessary to perform a set of calculations a specific number of times, and that number does not depend on the statements within the loop. Consider the problem of summing the first 20 integers:

$$SUM = \sum_{n=1}^{20} n$$

Such a problem is ideally suited to the FOR loop and could be evaluated using the following MAPOL program:

```
MAPOL
INTEGER N,SUM, TOP;
TOP := 20;
SUM := 0;
FOR N = 1 TO TOP DO
SUM := SUM + N;
ENDDO;
PRINT ("1X, 'SUM = ', I5)" , SUM );
END;
```

The general syntax of the FOR loop is

```
FOR <var> = <exp1> TO <exp2> [ BY <exp3> ] DO
.....
.....
ENDDO;
```

The loop counter <var> is called the control variable and may be any integer or real variable. <exp1>, <exp2> and <exp3> are called the initial, terminal and incremental parameters, respectively. Note the incrementation clause

BY <exp3>

is optional, as in the example. If it does not appear, the increment is taken to be one. Each loop terminates with the instruction ENDDO. The following rules must be noted:

- (1) If <exp1> > <exp2>, then the body of the loop will still be executed once.

- (2) The type of the control variable and the three expressions must be the same.
- (3) The control variable may not be redefined inside of the loop.

#### 4.3.2 The WHILE...DO Loop

Another way to execute a group of statements repeatedly is with a WHILE loop. This type of loop is used to repeat groups of statements that typically modify a more complex condition than the simpler incrementation of the FOR loop. As an example, suppose it is desired to compute the cube root of a number "X". If "a" is an approximation to the answer, then

$$b = \frac{2a + \frac{x}{a^2}}{3}$$

is an improved guess. The program of Figure B-2 will compute the cube root of 10 to 3 significant figures:

```

MAPOL
  REAL X,OLD,NEW,TEMP,EPS;
  X := 10;
  OLD := 2; $ THE INITIAL GUESS $
  NEW := 1;
  EPS := 0.001; $ THE CONVERGENCE CRITERION $
  WHILE ABS(OLD-NEW) > EPS DO
    TEMP := NEW;
    NEW := (2.0*OLD+X/OLD**2)/3.0;
    OLD := TEMP;
  ENDDO;
  PRINT ( " (1X,'X,CUBERT(X) ',2F15.5)" , X , NEW );
END;

```

Figure B-2. A MAPOL Program to Compute Cube Roots

The general form of the WHILE loop is:

```

  WHILE <cond> DO
    .....
    .....
  ENDDO;

```

The <cond> is any conditional expression that results in a logical outcome.

#### 4.4 THE IF STATEMENT

It is often necessary in a program to specify two or more alternatives that must be selected depending upon other program results. The IF statement allows this selection. There are three types of IF statements in MAPOL:

- o LOGICAL IF
- o BLOCK IF
- o IF ... THEN ... ELSE

##### 4.4.1 The Logical IF

The logical IF is used if a single expression is to be executed based on a particular condition. The syntax of this statement is

IF <cond> <statement>

where <cond> is any logical expression and <statement> is any legal executable MAPOL statement except:

- 1) A WHILE or FOR loop
- 2) Another logical IF
- 3) An END, ENDP, ENDIF, or ENDDO instruction
- 4) A PROC definition

Examples of the logical IF are:

```
IF A<B    PRINT("1X,'A = ',I5)" , A );
IF ABS(NEW-OLD) > EPX NEW := OLD;
IF A AND B OR C CALL UIMPRT ([,KMAT]);
```

##### 4.4.2 The Block IF

It is often necessary to perform a number of instructions based on a given condition. This can be accomplished by a block IF statement, the syntax of which is:

```
IF <cond> THEN
.....
.....
ENDIF;
```

Rather than a single statement, the body of the block may contain any number of MAPOL statements, for example:

```
IF A < B THEN  
    C := 1.0;  
    D := 4.0;  
    CALL UIMPRT (, [MMAT]);  
ENDIF;
```

#### 4.4.3 The IF...THEN...ELSE

The IF...THEN...ELSE statement is used to execute one of two separate blocks of code depending on a specific condition. The syntax of this statement is:

```
IF <cond> THEN  
    .....  
    .....    BLOCK 1  
    .....  
ELSE  
    .....  
    .....    BLOCK 2  
    .....  
ENDIF;
```

If the <cond> is satisfied, the instructions in BLOCK 1 are executed. If <cond> is not satisfied, then BLOCK 2 is executed.

#### 4.4.4 Nested IF Statements

IF statements may be nested to any level. That is, each IF or ELSE part may contain another IF statement, as shown below:

```
IF A > B THEN  
    A := 100;  
ELSE IF C < D THEN  
    C := 200;  
    ELSE  
    C := 0;  
    ENDIF;  
ENDIF;
```

Note that each IF...THEN...ELSE must terminate with its own ENDIF. It is helpful to indent code so that the blocks are obvious.

#### 4.5 THE END AND ENDP STATEMENTS

The END and ENDP statements are used to indicate the physical end of a MAPOL program or in-line procedure, respectively.

## SECTION V

### INPUT/OUTPUT STATEMENTS

#### 5.1 INTRODUCTION

The MAPOL compiler does not have facilities for input in the programming language. All input is handled by the ASTROS executive system. MAPOL does, however, allow direct output to the system print device as defined by the ASTROS host computer. Output is merged with the same file that contains all of the other ASTROS print output.

#### 5.2 THE PRINT STATEMENT

Output printing is requested with the PRINT statement, the syntax of which is:

```
PRINT ( <format> [, <print-list> ] );
```

In order to allow maximum power and flexibility while minimizing training, the <format> specifications used by MAPOL are identical to those used by FORTRAN. The format is entered as a literal string, enclosed by quotation marks; i.e.,

```
"(1X,5E1.6)"  
"(//1X,' X= ',F15.5)"
```

The <print-list> is a list of one or more defined variables to be printed. If only heading information is being printed, the <print-list> may be omitted. Examples of print statements are:

```
PRINT ("(1X,E15)",I,J,K);
```

which prints the three integer variables I, J and K using the indicated format and

```
PRINT ("(1X,'THIS IS A HEADER')");
```

which prints the message "THIS IS A HEADER".

ASTROS does not attempt to check the validity of a format statement with the data types being printed. As a result, it is possible to cause a FORTRAN run-time error condition.



## SECTION VI

### PROCEDURES AND FUNCTIONS

#### 6.1 INTRODUCTION

One of the most powerful features of a programming language is the ability to define "procedures," or subroutines, that perform specialized tasks. Some procedures with special characteristics are called "functions." Each MAPOL main program, procedure or function is called a "program unit." This section explains the use of procedures and provides examples of their use.

#### 6.2 PROGRAM UNITS AND SCOPE OF VARIABLES

In Section 1, a MAPOL program was defined very simply as having the form:

```
MAPOL
...
...
...
END;
```

This form is called a main program. A main program may also contain other program units that may be procedures or functions such as:

```
MAPOL
      PROC A;
      ...
      ...
      ENDP;
      REAL FUNC B;
      ...
      ...
      ENDP;
      ...
      ...
      END;
```

All procedures must appear in the main program before any executable statements, but after the declaration statements.

Each procedure, or function, may have variable declarations within it. If it does, these variables are called "local" to the procedure. Variables defined in the main program are called "global." These definitions are called the scope of the variables. The value of a local variable is not available outside of the procedure in which it is defined. Examples are shown after some additional definitions.

### 6.3 DEFINING A PROCEDURE

A procedure is defined in MAPOL by a declaration:

PROC <procname> [ <params> ];

where <procname> is any identifier. If this name is the same as a run-time procedure, the new procedure will be used. <params> is an optional list of formal parameters that are used to pass information into and retrieve information from the procedure.

<params> := ( <paramlist> )

<paramlist> := <ident> | <paramlist> , <ident>

Examples are:

PROC MYPROC ( A , B , C ) ;

PROC GETONE ;

The PROC statement is called the procedure "head". It is followed by the procedure "body" and an ENDP statement:

PROC TEST;

...  
 ...  
 ...  
 } PROCEDURE BODY  
ENDP;

This defines the procedure program unit. As an example, to find the square root of a real number

$$a = (b)^{1/2}$$

a Newton-Raphson iteration technique can be used

$$a_{n+1} = a_n - (a_n^2 - b)/2a_n$$

The iteration proceeds to the desired accuracy as determined by

$$| a_n - a_{n+1} | < EPS$$

A MAPOL procedure for this is shown in Figure 3.

```

PROC USQRT(A,SQRTA);
REAL A,SQRTA,EPS,DELTA,AOLD;
  EPS := 0.0001;
  SQRTA := 1.0;
  DELTA := 1.0;
  WHILE ABS(DELTA)>EPS DO
    AOLD := SQRTA;
    SQRTA := AOLD - ((AOLD*AOLD-A)/(2.0*AOLD));
    DELTA := SQRTA - AOLD;
  ENDDO;
ENDP;

```

Figure B-3. A MAPOL Procedure to Determine the Square Root of a Number

#### 6.4 INVOKING A PROCEDURE

Once procedures are defined, they may be used anywhere within the main program or in a subsequent procedure. This is done with the MAPOL statement:

```
CALL <procname> [ <userparm> ];
```

where <procname> is one of the defined procedures. The optional <userparm> are the actual user-defined variables to be passed to the procedure. They must agree in number and type with the PROC definition. Parameters are passed by name. For example, a program segment using the square root procedure of Figure 3 is

```

MAPOL
REAL X,Y;
....
X := 5;
CALL USQRT(X,Y);
....
END;

```

The parameters "X" and "Y" are the actual variables that will be used in place of the formal parameters in the procedure definition.

#### 6.5 FUNCTION PROCEDURES

A special kind of procedure that can have only one output value is called a FUNCTION. Because it is a value, the type of the function must be a

declared. Valid types are integer, real, complex or logical. Therefore, the function head differs slightly from that of the procedure:

`<type> FUNC <funcname> [ <params> ]`

Again, `<type>` must be included and all other rules are the same as those for a regular procedure.

Unlike procedures, functions are invoked with their name and arguments as in FORTRAN and they can, therefore, be used directly in assignment statements and expressions, e.g.,

`A := SIN(X);`  
`B := X + Y * SQRT(Z);`

#### 6.6 INTRINSIC FUNCTION PROCEDURES AND INTRINSIC PROCEDURES

In addition to the user defined procedures and functions within a MAPOL main program unit, MAPOL provides a set of predefined functions and procedures to perform certain tasks in a similar manner to other high level languages such as FORTRAN. These "intrinsic" procedures are in addition to the engineering modules defined as part of the ASTROS system generation process. The set of intrinsic procedures within the MAPOL language can be broken into three groups: intrinsic mathematical functions, intrinsic relational procedures and general intrinsic procedures. Each group is discussed separately in the following sections.

#### 6.7 INTRINSIC MATHEMATICAL FUNCTIONS

Table B-11 shows the list of intrinsic mathematical functions available in MAPOL. These functions make up the mathematical function library within the MAPOL language and provide the user with the capacity to perform a wide variety of tasks within the MAPOL program units. With very few exceptions, the MAPOL mathematical functions are identical in form to those in the FORTRAN language; the exceptions are noted in Table B-11. Trigonometric functions in MAPOL use radian angles as arguments and result in radian angles just as in FORTRAN. All MAPOL functions are "generic" in the sense that they support multiple data types (INTEGER, REAL) as arguments and perform the appropriate conversions.

**TABLE B-11. INTRINSIC MATHEMATICAL FUNCTIONS IN MAPOL**

NAME	DESCRIPTION	EXAMPLE
ABS	Absolute Value	A := ABS( B );
ACOS	Arccosine	A := ACOS( B );
ASIN	Arcsine	A := ASIN( B );
ATAN	Arctangent	A := ATAN( B );
COS	Cosine	A := COS( B );
COSH	Hyperbolic Cosine	A := COSH( B );
EXP	Exponential	A := EXP( B );
IMAG	Imaginary Component (Equivalent to FORTRAN AIMAG)	A := IMAG( B );
LN	Natural Logarithm (Equivalent to FORTRAN LOG)	A := LN( B );
LOG	Common Logarithm (Equivalent to FORTRAN LOG10)	A := LOG( B );
MAX	Selects Largest Value	A := MAX( B, C, ... );
RE	Real Component (Equivalent to FORTRAN REAL)	A := RE( B );
SIN	Sine	A := SIN( B );
SINH	Hyperbolic Sine	A := SINH( B );
SQRT	Square Root	A := SQRT( B );
TAN	Tangent	A := TAN( B );
TANH	Hyperbolic Tangent	A := TANH( B );

#### 6.8 INTRINSIC RELATIONAL PROCEDURES

As discussed in Section 2.4, MAPOL has provided a means by which individual relational entries (row/attribute combinations) may be accessed directly. Table B-12 shows the argument lists to the set of intrinsic procedures provided to enable the MAPOL programmer to open relations, to retrieve particular rows, to update or add rows and to close the relation. In combination with the RELATION and PROJECT declarations, these procedures provide a direct data base interface that nearly matches the full relational application programming interface in CADDB. The equivalent CADDB application interface routine is listed in Table B-12 for completeness. There is an implementation maximum of five open relational variables at any time during the execution of a MAPOL program.

**TABLE B-12. INTRINSIC RELATIONAL PROCEDURES IN MAPOL**

##### A. RECEND

Calling Sequence: CALL RECEND ( <rel-var> );

Purpose: To end the definition of relational conditions. (A maximum of 10 may be applied per relation).

CADDB equivalent: REENDC

##### B. RELCND

Calling Sequence: CALL RELCND ( <rel-var>, <attr>, <relop>, <value> );

**TABLE B-12. INTRINSIC RELATIONAL PROCEDURES IN MAPOL (Continued)**

Purpose: To define relational conditions.

<attr> is an attribute name in quotation marks

<relop> is one of "GT", "LT", "EQ", "NE", "GE", "LE"

<value> is the conditional value

(A maximum of 10 may be applied per relation)

CADDB equivalent: RECOND

**C. RELADD**

Calling Sequence: CALL RELADD ( <rel-var> );

Purpose: To add a tuple to a relation.

CADDB equivalent: READD

**D. RELEND**

Calling Sequence: CALL RELEND ( <rel-var> );

Purpose: To close a relation.

CADDB equivalent: DBCLOS

**E. RELGET**

Calling Sequence: CALL RELGET ( <rel-var>, <status> );

Purpose: To retrieve a tuple from an open relation.

<status> is an integer variable that is non-zero if an error occurred.

CADDB equivalent: REGET

**F. RELUPD**

Calling Sequence: CALL RELUPD ( <rel-var> );

Purpose: To replace a previously retrieved relational tuple with modified values.

CADDB equivalent: REUPD

**G. RELUSE**

Calling Sequence: CALL RELUSE ( <rel-var>, <ntuple>, <status> );

Purpose: To open a relation.

<ntuple> is an integer variable which contains the number of tuples in the relation on output.

<status> is an integer variable that is non-zero if an error occurred.

CADDB equivalent: DBOPEN

Figure B-4 shows a simple MAPOL procedure that manipulates a relation called GPOINT. Two rows are placed in the relation followed by a conditional retrieval to obtain one of the tuples for use in an additional operation.

```

MAPOL
RELATION GPOINT;
INTEGER GI, NIUPLES, ERRSTAT;
REAL    X, Y,      Z,      SUM;
$
PROJECT GPOINT USING GID, X, Y, Z;
$
CALL RELUSE ( GPOINT, NIUPLES, ERRSTAT );
PRINT ("(' NIUPLES = ', I6)", NIUPLES);
IF ERRSTAT <> 0 THEN
    PRINT ("(' ERROR STATUS ',I5,' IN OPENING GPOINT')",
           ERRSTAT );
    CALL EXIT;
ENDIF;
$
    DEFINE THE FIRST GPOINT ROW
$
GPOINT@GID := 1;
GPOINT@X   := 5.0;
GPOINT@Y   := 6.0;
GPOINT@Z   := 7.0;
CALL RELADD ( GPOINT );
$
    DEFINE THE SECOND GPOINT ROW
$
GPOINT@GID := 5;
GPOINT@X   := 15.0;
GPOINT@Y   := 16.0;
GPOINT@Z   := 17.0;
CALL RELADD ( GPOINT );
CALL RELADD ( GPOINT );
$
    SET THE CONDITION TO RETRIEVE THE SECOND TUPLE
$
CALL RELUSE ( GPOINT, NIUPLES, ERRSTAT );
CALL RELCND ( GPOINT, "GID", "GT", 2);
CALL RECEND ( GPOINT );
CALL RELGET ( GPOINT, ERRSTAT );
$
    SUM UP THE COORDINATES OF THE SECOND GRID
$
SUM := GPOINT@X + GPOINT@Y + GPOINT@Z;
PRINT ("(' SUM OF X+Y+Z IS = ', F10.5)", SUM);
END;

```

Figure B-4. MAPOL Program Using Relational Procedures

## 6.9 GENERAL INTRINSIC PROCEDURES

Two other intrinsic procedures have been provided to enhance the utility of MAPOL: the EXIT and TRNSPOSE procedures. The first is identical to the common FORTRAN extension EXIT. The MAPOL statement

```
CALL EXIT;
```

will cleanly terminate the ASTROS execution without requiring the user to

jump to the end of the MAPOL sequence. This is particularly useful when an edited standard solution sequence is used. The TRNSPOSE procedure provides an additional MATRIX operation that is otherwise missing from the language. While the operation

$$[A] := \text{TRANS}(B) * [C];$$

is available within the syntax of MAPOL expression, the operation

$$[A] := \text{TRANS}(B);$$

is not. The intrinsic procedure TRNSPOSE allows this matrix operation to be performed. The form of TRNSPOSE is

$$\text{CALL TRNSPOSE} ([A], [\text{TRANSA}]);$$

where [A] is the matrix to be transposed and [TRANSA] is the resultant transposed matrix.



## APPENDIX C

### ANNOTATED STANDARD EXECUTIVE SEQUENCE

This appendix contains a description of the standard executive (MAPOL) sequence for the ASTROS system. This single MAPOL program is capable of performing any and all of the ASTROS system capabilities. As a result, the sequence is quite long (over 1500 lines) and appears very complicated. Study of the MAPOL sequence will reveal, however, that many of the program structures are used repeatedly throughout the sequence. The purpose of this appendix is to provide a familiarity with the standard sequence sufficient to make modifications to that sequence. This documentation does not take the place of the Programmer's Manual, but is useful in that the relationships among the engineering modules and the matrix expressions in the standard sequence are shown. Thus, the user can more readily identify where particular quantities that may be of interest are computed and stored.

The principal MAPOL statements in the standard sequence are identified by line number and described in some detail. Line numbers that do not appear in this presentation are omitted because it is felt that the statement or statements are sufficiently readable in the context of the surrounding code. Program blocks that appear repeatedly will be described each time to enhance readability and for completeness. The MAPOL sequence is self-documented with comments and indentation is used to identify all block program structures. These features are helpful in identifying the top of major program blocks and in following the structure of the program. In general, it is felt that, despite its seeming complexity, the MAPOL sequence that directs the ASTROS procedure is understandable and much more readable than its NASTRAN counterparts.

The user is cautioned that, due to ongoing modification, the standard sequence shown in this appendix and used for this discussion may NOT be identical to the actual standard sequence. The sequence included will be very similar and should be sufficient for discussion. It is imperative that the user wanting to modify the standard sequence use the line numbers that appear in the standard sequence listing contained in the system generation (SYSGEN) output rather than those in this appendix.

All line numbers reference those shown in the standard MAPOL sequence listing of Figure C-1. The phrase "matrix [A] is equivalenced to matrix [B]" is used frequently in this discussion. This denotes a special MAPOL feature in which the operation "[A]:-[B];" is performed using the CADDB DBEQUV utility rather than an actual copy operation taking place.

#### C.1 VARIABLE DECLARATIONS

LINE 2. Identification line showing the last modification date of the standard sequence. If the user's SYSGEN output has a different date, the sequence shown differs from the actual sequence to some degree.

LINES 4 TO 180. The MAPOL language requires that all variables appearing in the MAPOL sequence be declared. Every scalar and high order variable that appears anywhere in the sequence is declared in these lines. The declarations are grouped by association with a particular module or discipline. Each such group is labeled with a leading comment.

NOTE: User defined MAPOL procedures would be inserted into the standard sequence following Line 180.

#### C.2 PREFACE MODULES

LINE 190. Calls the Solution Control interpreter to process the SOLUTION packet and to load the CASE relation with requested disciplines and subcases.

LINE 191. Calls the Input File Processor module to process the BULK DATA packet, to load the input data onto the data base and to generate basic collections of data including the transformation matrices (TMAT relation) and the basic nodal data (BCPDT relation).

LINES 192 TO 195. Make calls to the structural set definition module, MKUSET, for each boundary condition in the Solution Control. This module fills the USET entity and creates the partitioning vectors, [Pxxx], and vectors of enforced displacement, [YS], needed to perform matrix reductions of the structural matrices.

LINE 199. Calls the element summary table generation module, MAKEST, to form the element summary data for each structural element in the model. Preliminary design variable linking is also performed in this module.

LINE 200. Calls the element sensitivity matrix generation module, EMG. This is the fundamental element processing module in ASTROS. It generates all the local design variable (element) sensitivities for stiffness, mass, thermal loads and stress and/or strain constraints (KELM, MELM, TELM, SMAT, respectively) for all the structural elements.

LINE 201. Calls the generic bulk data preface module, PFBULK. This module performs preliminary error checking for a number of disciplines in order to avoid costly error checking within the design iteration loop. Preliminary collections of data are also formed in some cases.

LINE 206. Calls the first phase element sensitivity matrix assembly module, EMAl. This module assembles the data from EMG into the global design sensitivities for stiffness and mass (DKVI and DMVI, respectively).

LINE 211. Calls the loads sensitivity assembly module, LODGEN. This module assembles the element thermal loads sensitivities from EMG (TELM), the global mass sensitivities from EMAl (DMVI) and the design invariant loads from the bulk data (FORCE, MOMENT, etc.) into the global simple loads sensitivities for all loads defined in the bulk data. Final loads assembly occurs in the subsequent boundary condition loops in the GTLOAD module.

LINE 216 TO 223. Call the preface aerodynamics routine, PFAERO, to compute the aerodynamic matrices for both steady and unsteady aerodynamics. The call appears in a loop which is controlled by PFAERO. The module will be called once for each Mach number used in steady aeroelastic disciplines in the Bulk Data.

The AMP module performs additional operations on the unsteady aerodynamic matrices. These are separated from PFAERO because the PFAERO operations are purely geometrical while the AMP operations are a function of the particular Mach numbers and reduced frequencies requested in the Bulk Data. The user may, therefore, choose to perform PFAERO in an initial run and AMP in subsequent restart runs.

### C.3 OPTIMIZATION PHASE

LINE 229. This IF test checks for the existence of any optimization boundary conditions in the Solution Control. If none are present, control jumps to the start of the ANALYSIS phase at Line 1040.

LINE 230 TO 237. Initialize "constants" that are used in the optimization phase. Of particular importance to Math Programming methods is MOVLIM, which specifies the maximum move in a single iteration. ALPHA plays the same role for Fully Stressed Design (FSD) and MAXFSD specifies the maximum number of FSD cycles that are performed before reverting to Math Programming. MAXITER and CONVRGLIM control the maximum number of all iterations and set the percent change global convergence criteria, respectively, which are applicable to both optimization methods.

#### C.4 DESIGN CONVERGENCE LOOP

LINE 238. Initializes the convergence loop for the design iterations. Once global convergence is reached (CONVERGE=2) or the maximum number of iterations has been performed, control is passed to the start of the ANALYSIS phase at Line 1040.

LINE 242. Increments the iteration counter, NITER.

LINE 243. Calls the thickness constraint evaluation module, TCEVAL. This module not only evaluates any thickness constraints, but also initializes several entities which need to be flushed between design iterations.

LINE 244. Calls the global matrix assembly module, EMA2. This module forms the stiffness and mass matrices in the structural set for the current design.

#### C.5 OPTIMIZATION PHASE BOUNDARY CONDITION LOOP

LINE 248. Initiates the analysis boundary condition loop for the OPTIMIZATION phase. After all (optimization) boundary conditions have been completed, control passes to active constraint selection and sensitivity calculation at Line 758. Note that matrices that are required for sensitivity calculations are subscripted BC in the MAPOL program, e.g., [GSUBO(BC)].

LINE 249 TO 252. Call the BOUND and BDCASE modules which determine the nature of the current boundary condition. Parameters are computed and passed to the MAPOL sequence defining the disciplines and matrix reductions to be performed. The parameters have the following definitions:

NMPC, NSPC, NOMIT, NRSET NGDR	The number of dependent degrees of freedom in each structural set. Set to -1 by BOUND if dynamic reduction is used in the current boundary condition.
BLOAD	Set to 1 if there are any static loads in the boundary condition.
BMASS	Set to 1 if the mass matrix is required by any discipline in the boundary condition.
BMODES	Set to 1 if any discipline in the boundary conditions requires a normal modes analysis.
BSAERO	Set to 1 if there are any static aeroelastic analyses in the current boundary condition.
QDP	Dynamic pressure of the SAERO subcase(s).
MINDEX	Index for the appropriate steady aerodynamic matrices
SYM	Symmetry flag for SAERO; - 1 if symmetric - 2 if antisymmetric.
TRMTYP	Type of trim for any SAERO disciplines; - 0 if no trim - 1 if lift only trim - 2 if lift/pitching moment trim
BFLUTR	Set to 1 if any flutter analyses in the current boundary condition.
BDYN	Set to 1 if any dynamic response disciplines (Except BLAST).
BDTR, BMTR	Set to 1 if any (D)irect or (M)odal (TR)ansient response analyses, respectively.
BDFR, BMFR	Set to 1 if any (D)irect or (M)odal (FR)equency response analyses, respectively.
BGUST	Set to 1 if any GUST options are selected in the current boundary condition.
BBLAST	Set to 1 if any BLAST analyses are in the current boundary condition.

Note that all subsequent operations within the analysis boundary condition loop will be directed by these flags.

LINE 253. Calls the GTLOAD module to obtain the applied static loads representing the current design from the design load sensitivities and simple loads formed in LODGEN.

LINES 254 TO 262. Equivalence the proper aerodynamic influence coefficient matrix for steady aeroelastic analyses to the generic [AIC] matrix.

LINES 266 TO 292. This program block performs the required matrix reductions for Multipoint Constraints for all matrices that must be reduced, as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KGG], mass matrix [MGG], the static loads matrix [PG], SAERO splining

matrices [GTKG] and [GSTKG], and the unsteady aerodynamic splining matrix [UGTKG]. If no reductions are required, the n-set matrices are equivalenced to the g-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 293 TO 320. This program block performs the required matrix reductions for Single Point Constraints for all matrices that must be reduced as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KNN], mass matrix [MNN], the static loads matrix [PN], SAERO splining matrices [GTKN] and [GSTKN], and the unsteady aerodynamic splining matrix [UGTKN]. If no reductions are required, the f-set matrices are equivalenced to the n-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 324 TO 331. This program block transforms the steady aerodynamic influence coefficient matrix [AIC] and the air loads matrix [AIRFRC] to the structural nodes using the SAERO spline matrices. The stiffness matrix [KAFF] is then formed which includes the aeroelastic stiffness based on [AICS] and QDP.

LINES 333 TO 432. This program block performs the required matrix reductions to the analysis set for all matrices that must be reduced as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KFF], mass matrix [MFF], the static loads matrix [PF], and the unsteady aerodynamic splining matrix [UGTKF]. There are three possible paths through this block of MAPOL code:

#### GENERAL DYNAMIC REDUCTION (LINES 339 TO 372)

The modules GDR1, GDR2, GDR3 and GDR4 are called to compute the necessary transformation matrix [GSUBO]. This matrix is then used to perform the matrix reductions.

#### STATIC REDUCTION (LINES 374 TO 418)

There are two subpaths through this block depending on whether the stiffness matrix is symmetric or asymmetric (which is the case for SAERO disciplines).

#### ASYMMETRIC STATIC REDUCTION (LINES 379 TO 398)

This block of code is used only for SAERO disciplines and assumes that there are support degrees of freedom (as is required for SAERO analyses). Lines 379 to 385 perform some operations on the symmetric stiffness matrix to ensure that the quantities needed to compute the [D] matrix for later support set reduction are available. Since the [D] matrix is design invariant, it is only computed on the first iteration. Lines 386 to 388 reduce the asymmetric stiffness matrix to the analysis set. Note that the off diagonal partition [KOA], the asymmetric decomposed stiffness matrices [KOOU] and [KOOL] and the air loads on the omitted degrees of freedom [POARO] are output from FREDUCE for use in the recovery and design sensitivity operations. Lines 392 to 398 reduce the mass matrix to the analysis set. Note that the [IFM] matrix is computed for use in the support set reduction and sensitivity operations.

#### SYMMETRIC STATIC REDUCTION (LINES 400 TO 418)

This block of code is used for all disciplines other than SAERO using static reduction. Lines 400 to 401 reduce the symmetric stiffness matrix to the analysis set. Note that the symmetric decomposed stiffness matrices [KOOINV] and the static loads on the omitted degrees of freedom [PO] are output from FREDUCE for use in the recovery and design sensitivity operations. Lines 405 to 411 reduce the mass matrix to the analysis set. Note that if there are also support degrees of freedom, the [IFM] matrix is computed for use in the support set reduction and sensitivity operations.

Lines 413 to 418 reduce the unsteady aerodynamic spline matrices to the analysis set. The roundabout order of operations is dictated by efficiency considerations for the MPYAD operation.

#### NO REDUCTION (LINES 423 TO 431)

If no reductions are required, the a-set matrices are equivalenced to the f-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 434 TO 527. This program block performs four functions: (1) support set reduction of the analysis set matrices; (2) static solution (either SAERO or STATICS); (3) normal modes analysis; and (4) recovery of displacements and accelerations to the analysis set.

SUPPORT SET REDUCTION (LINES 438 TO 507)

Lines 439 to 441 compute the design invariant [D] matrix on the first iteration. On subsequent iterations, Lines 444 to 445 decompose the 1-set stiffness matrix if static loads are present in the boundary condition (note that [KLLINV] is not needed for other disciplines after the [D] matrix has been computed). Lines 451 to 456 compute the reduced mass matrix [MRR] and other matrices needed for inertia relief computations: [IFR], [R22], and [R21].

Lines 461 to 481 perform the steady aeroelastic analyses in the current boundary condition. Lines 476 to 477 call the SAERO module which performs the trim calculation and evaluates aeroelastic effectiveness constraints. The matrix [DELTA] contains the vector of trim parameters on output. Note that the trimmed omitted loads [PO] are computed from the [POARO] and [DELTA] matrices. The displacements and accelerations are recovered to the analysis set in Lines 478 to 480.

Lines 488 to 502 perform static analysis with inertia relief. The INERTIA module is called on Line 497 to compute the accelerations on the support points [AR]. The displacements and accelerations are then recovered to the analysis set in Lines 498 to 501.

Lines 504 to 505 perform real eigenvalue extraction including rigid body modes. The eigenvectors are output in the [PHIA] matrix and the LAMBDA relation contains the eigenvalue extraction data. Also output are the generalized mass matrix, [MII], and the number of extracted eigenvectors, HSIZE. Line 506 calls the frequency constraint evaluation module.

NO SUPPORT SET REDUCTION (LINES 512 TO 527)

In this case, all the computations are performed in the analysis set. Lines 513 to 514 solve for displacements due to static applied loads. Lines 516 to 522 represent a block of code needed for the currently nonfunctional capability to perform static aeroelastic



analyses with input trim parameters. Lines 524 to 525 perform real eigenvalue extraction without rigid body modes and frequency constraint evaluation, respectively. The eigenvectors are output in the [PHIA] matrix and the LAMBDA relation contains the eigenvalue extraction data. Also output are the generalized mass matrix, [MII], and the number of extracted eigenvectors, HSIZE.

LINES 532 TO 550. This program block performs the requested dynamic response disciplines except for BLAST. Note that these analyses are included in the OPTIMIZE phase only for completeness: they do not generate any design constraints.

Lines 533 to 534 call the QHHLGEN module to perform the design dependent modal reduction of the unsteady aerodynamic matrices for FLUTTER and GUST analyses to the dynamic set (including extra point degrees of freedom). Outputs are the [PHIKH] transformation matrix of normal modes at aerodynamic boxes, the [QHHL] matrix list for flutter and the [QHJL] matrix list for gust.

Lines 535 to 539 generate the stiffness, mass and damping matrices in the dynamic set (including extra point degrees of freedom) from DMA and the applied loads at time or frequency steps from DYNLOAD. [PDT] contains the direct or modal applied loads at time steps and [PDF] the direct or modal applied loads at frequency steps.

Lines 540 to 541 perform modal flutter analysis using the [KHHF] stiffness matrix, and the [MHH] generalized mass matrix from DMA and [QHHL] from QHHLGEN. The applied flutter constraints are also evaluated in this module.

Lines 543 to 549 perform any TRANSIENT or FREQUENCY response analyses using the matrices generated in DMA and DYNLOAD. The outputs are either modal, [UxxxxI], or physical, [UxxxxA], and extra point [UxxxxE] displacements, velocities and accelerations. If modal analyses were performed, the physical set responses are computed in Lines 547 to 548.

LINES 552 TO 578

This program block performs BLAST analyses. Note that these analyses are included in the OPTIMIZE phase only for completeness: they do not generate any design constraints. The BLASTFIT module converts the unsteady aerodynamics matrices from the frequency domain to the time domain and also provides

a number of matrices that are used in setting up a trim analysis. Lines 555 through 557 replace the rigid body modes calculated by REIG with the physical modes contained in the [D] matrix and Lines 558 through 562 set up generalized matrices (i.e., mass, stiffness, and aerodynamic). The BLASTRIM module at Line 573 performs a trim analysis prior to the blast encounter to set up initial conditions. The BLASTDRV module then calculates the aircraft response as a function of time. The outputs are the modal displacements, velocities and accelerations at the requested time steps in the matrix [UBLASTI].

LINES 582 TO 644. This program block performs the required matrix recovery from the analysis set (a-set) to the free degrees of freedom (f-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the statics displacement matrix [UF], the acceleration matrix [AF], the eigenvectors [PHIF] and the transient and frequency physical response matrices [UTRANF] and [UFREQF]. There are three possible paths through this block of MAPOL code:

RECOVERY WITH GENERAL DYNAMIC REDUCTION (LINES 587 TO 597)

The [GSUBO] transformation matrix computed during matrix reduction is used to recover the response quantities at the free degrees of freedom.

RECOVERY WITH STATIC REDUCTION (LINES 603 TO 627)

Like the corresponding reduction, there are two recovery paths depending on the symmetry of the stiffness matrix. Lines 604 to 607 recover the displacements and accelerations using the symmetric [KOOINV] and [GSUBO] matrices computed in the symmetric reduction path. Lines 610 to 614 recover the displacements and accelerations using the asymmetric decomposed omitted stiffness partitions [KOOU] and [KOOL] and [GASUBO] computed in the asymmetric path. Lines 616 to 617 perform recovery operations for normal modes, transient response and frequency response disciplines, all of which require that symmetric stiffness reduction was performed.

RECOVERY WITH NO REDUCTIONS (LINES 423 TO 431)

If no reductions were applied, the f-set matrices are equivalenced to the a-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 646 TO 682. This program block performs the required matrix recovery from the free degrees of freedom (f-set) to the independent set (n-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the statics displacement matrix [UN], the acceleration matrix [AN], the eigenvectors [PHIN] and the transient and frequency physical response matrices [UTRANN] and [UFREQN]. If no reductions were applied, the n-set matrices are equivalenced to the f-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 684 TO 732. This program block performs the required matrix recovery from the independent set (n-set) to the global structural set (g-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the statics displacement matrix [UG], the acceleration matrix [AG], the eigenvectors [PHIG] and the transient and frequency physical response matrices [UTRANG] and [FREQG]. If no reductions were applied, the g-set matrices are equivalenced to the n-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINE 736. The modal blast responses are recovered to the physical set.

LINES 740 TO 742. Call the DCEVAL module to evaluate any displacement constraints that were applied to any of the STATICS or SAERO subcases. If either STATICS or SAERO disciplines exist, the SCEVAL module is called to evaluate applied stress and/or strain constraints. Note that the STATICS and SAERO disciplines are mutually exclusive within one boundary condition.

LINES 746 TO 753. This program block handles output requests not handled directly within the analysis modules. The OFPLOAD module satisfies applied load output requests and the OFPDISP module satisfies displacement, velocity, acceleration and real eigenvalue (ROOTS) output requests. The EDR module computes requested element response quantities (e.g., stress and strain) and the computed quantities are printed in OFPEDR.

LINE 754. This statement marks the end of the analysis boundary condition loop of the optimization phase of the standard MAPOL sequence.

## C.6 ACTIVE CONSTRAINT SELECTION

LINE 758 TO 761. Set the constraints screening parameters NRFAC and EPS and call the ACTCON module which selects the constraints to be retained in the Math Programming optimization path. ACTCON also checks on the convergence of the actual design problem if the approximate problem was designated as converged. Final convergence requires that the maximum constraint value lie between bounds specified by the CTL and CTLMIN parameters.

LINE 763. This IF block checks to see if global convergence or the maximum number of iterations has been reached. If so (CONVERGE = 2 or NITER > MAXITER), the optimization phase is complete and control jumps to the start of the analysis phase at Line 1040. The MAPOL program is set up in this way to ensure that the last analysis performed in the optimization phase represents the converged design. Sensitivity analysis is not performed following global convergence.

LINE 767 TO 768. Call the FSD module to perform Fully Stressed Design optimization. If no FSD was selected in the Solution Control packet or if the number of iterations exceeds the MAXFSD parameter, the FSDFLG is set to zero to enable Math Programming to occur. A nonzero FSDFLG is returned if the FSD module performed a redesign based on the applied stress constraints.

LINE 770. The FSDFLG parameter output from the FSD module is checked. If it is zero, Math Programming is selected and the sensitivity portion of the optimization phase is entered. If nonzero, control passes to the top of the optimization phase at Line 229.

## C.7 SENSITIVITY EVALUATION FOR MATH PROGRAMMING

LINE 776. Calls the MAKDFV module to evaluate the sensitivities of boundary condition independent constraints that are explicit functions of the global design variables. These are currently limited to thickness constraints which are automatically generated when shape function design variable linking is used. The sensitivities are stored in the matrix [AMAT].

LINE 782. Initiates the sensitivity boundary condition loop for the OPTIMIZATION phase. After all (optimization) boundary conditions have been completed, control passes to the Math Programming optimization module DESIGN at Line 1028.

LINE 783 TO 785. Call the ABOUND module to determine whether the current boundary condition has active boundary condition dependent constraints. This enables ASTROS to avoid expensive sensitivity computations for inactive boundary conditions. Parameters are computed and passed to the MAPOL sequence defining the sensitivity evaluations and matrix reductions to be performed. The parameters have the following definitions:

ABC	Set to 1 if the boundary condition contains active constraints.
NAU	The number of displacement vectors for which there are active displacement dependent constraints. This initially includes one displacement vector for each active lift effectiveness constraint and two vectors for each active aileron effectiveness constraint. These pseudo displacement vectors are partitioned out in the AEROSENS module.
NACSD	The number of active stress and displacement constraints.
ADC	The number of active frequency constraints.
AFC	The number of active flutter constraints.
TRMTYP	Type of trim for any active SAERO disciplines; - 0 if no trim - 1 if lift only trim - 2 if lift/pitching moment trim.
MINDEX	Pointer to appropriate steady aeroelastic stability coefficients for active effectiveness constraints.
NAE	The number of aeroelastic effectiveness constraints.
NAUE	The number of active pseudo displacement fields associated with effectiveness constraints.
NMPC, NSPC, NOMIT, NRSET,	The number of dependent degrees of freedom in each structural set.
NGDR	Set to -1 by ABOUND if dynamic reduction is used in the current active boundary condition.

Note that all subsequent sensitivity operations within the sensitivity boundary condition loop will be directed by these flags.

LINE 786. The active boundary condition flag, ABC, is checked. If the boundary condition is not active ( $ABC = 0$ ), control jumps to the next boundary condition at Line 783. If no further boundary conditions, control jumps to the DESIGN module at Line 1028.

LINE 790. The sensitivities of any active frequency constraints are computed.

LINE 794 TO 796. The sensitivities of any active flutter constraints are computed.

LINE 797. This IF test checks for the existence of any active displacement dependent constraints (stress, displacement and aeroelastic effectiveness constraints). If none are present, control jumps to the top of the sensitivity boundary condition loop at Line 782. The block terminates at Line 1019.

LINES 801 TO 812. This program block computes the sensitivity of active stress/strain and displacement constraints to the displacements in the MAKDFU module. There are two methods of sensitivity evaluation provided for greater efficiency: the gradient method and the virtual load method. The gradient method requires that one forward backward substitution (FBS) be done for each active displacement for each design variable. The virtual load method requires one FBS for each active stress/strain or displacement constraint. The method requiring the fewest FBS operations is used. When the gradient method is used, the output from MAKDFU is stored in [DFDU], otherwise, it is stored in [DPGV].

LINE 817. The displacement vectors [UG] for the current active boundary condition are partitioned into the active and inactive vectors using the [PGA] partitioning vector computed in ABOUND. The inactive vectors are discarded, the active vectors are placed in [UGA].

LINE 822. The sensitivities of the active design dependent loads (if any) are computed in the DDLOAD module. If no design dependent loads exist, DDFLG is set to 0, otherwise it is 1. The sensitivities for all active load conditions, including design invariant subcases, are stored in the [DPVJ] matrix if DDFLG equals 1.

LINE 824. The matrix product of the stiffness matrix sensitivities (DKVI from EMAL) times the active displacements is computed in the MAKDVU module.

LINES 825 TO 833. This program block computes additional terms for inertia relief problems (both SAERO and STATICS). Line 828 computes the matrix product of the mass matrix sensitivities (DMVI from EMAL) times the active accelerations while Line 830 computes the matrix product of the mass matrix sensitivities times the active displacements. The matrix DUG is formed as the sum of DKUG and DMAG. If there are no support degrees of freedom, the

generic DUG matrix is merely equivalenced to the DKUG matrix from Line 824 at Line 832.

LINES 837 TO 855. This program block adds the loads sensitivities to the final [DUG] matrix resulting from MAKDVU module call(s). The result is placed in [DPGV] for the gradient method or [DFDU] for the virtual load method.

LINES 859 TO 867. This program block performs the matrix reductions for Multipoint Constraints for the [DPGV] matrix, as directed by the ABOUND flags. For active steady aeroelastic analyses or statics with inertia relief, the [DMUG] matrix is also reduced. If no reductions are required, the n-set matrices are equivalenced to the g-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 869 TO 877. This program block performs the matrix reductions for Single Point Constraints for the [DPNV] matrix, as directed by the ABOUND flags. For active steady aeroelastic analyses or statics with inertia relief, the [DMUN] matrix is also reduced. If no reductions are required, the f-set matrices are equivalenced to the n-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 879 TO 908. This program block performs the matrix reductions for the [DPFV] matrix to the analysis set as directed by the ABOUND flags. For active steady aeroelastic analyses or statics with inertia relief, the [DMUF] matrix is also reduced. There are three paths: Lines 881 and 882 perform reductions using the matrices computed in the analysis boundary condition loop for GDR; Lines 885 to 903 perform reductions using the asymmetric (886 to 894) or symmetric (896 to 902) reductions using the matrices computed in the analysis boundary condition loop for static condensation; Lines 905 to 906 perform the equivalences required in the case where no reductions are needed. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 910 TO 949. This program block performs four functions: (1) support set reduction of the analysis set matrices; (2) aeroelastic effectiveness constraint sensitivity calculations; (3) displacement sensitivity evaluation either with or without inertia relief; and (4) recovery of displacement sensitivities to the analysis set.

SUPPORT SET REDUCTION/EFFECTIVENESS SENSITIVITIES (LINES 911 TO 943)

Lines 911 to 915 compute matrices needed for inertia relief computations: [DP] and [DRHS].

Lines 920 to 932 perform the steady aeroelastic sensitivity evaluations. AEROSSENS computes the actual sensitivities for aeroelastic effectiveness constraints and the displacement and trim parameter sensitivities for stress/strain and displacement constraints associated with SAERO analyses in the current active boundary condition.

Lines 937 to 942 perform the sensitivity analysis for static analyses with inertia relief.

NO SUPPORT SET REDUCTION (LINE 948)

In this case, all the computations are performed in the analysis set. Line 948 performs the sensitivity computations if no support set reduction is required.

LINE 958. An additional check is made on the NAU parameter which may have been decremented by AEROSSENS due to the presence of pseudo displacements for aeroelastic effectiveness sensitivity calculations. If no real displacement fields are left, control jumps to the next boundary condition at Line 782. The IF block terminates at Line 1018. If any active strength constraints remain, the recovery phase of the sensitivity boundary condition loop is entered.

LINES 962 TO 988. This program block performs the required matrix recovery from the analysis set (a-set) to the free degrees of freedom (f-set) for the [DUFV] matrix as directed by the ABOUND flags. There are three possible paths through this block of MAPOL code:



#### RECOVERY WITH GENERAL DYNAMIC REDUCTION (LINE 964)

The [GSUBO] transformation matrix computed during matrix reduction is used to recover the response quantities at the free degrees of freedom.

#### RECOVERY WITH STATIC REDUCTION (LINES 966 TO 984)

Like the corresponding reduction, there are two recovery paths depending on the symmetry of the stiffness matrix. First, Lines 969 and 971 account for aeroelastic or static inertia relief effects, respectively. Then the sensitivities of omitted displacements to omitted loads are computed in Lines 977 to 979 or Lines 981 to 982 for SAERO and other disciplines, respectively. Line 984 then performs the merge operation to obtain the corrected analysis set displacement sensitivities.

#### RECOVERY WITH NO REDUCTIONS (LINE 986)

If no reductions were applied, the f-set matrices are equivalenced to the a-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous active boundary conditions.

LINES 992 TO 996. This program block performs the matrix reductions for Multipoint constraints for the left hand side matrix for the final sensitivity evaluation. This is the matrix stored in [DFDUN]. If no reductions are required, the n-set matrices are equivalenced to the g-set matrices.

LINES 998 TO 1002. This program block performs the matrix reductions for Single Point Constraints for the left hand side matrix for the final sensitivity evaluation. This is the matrix stored in [DFDUF]. If no reductions are required, the f-set matrices are equivalenced to the n-set matrices.

LINES 1006 TO 1016. Perform the final sensitivity computations in which the design sensitivities for all active strength constraints are computed. Separate paths are provided for the virtual load and gradient methods. Both paths use the MKAMAT module since identical matrix operations are performed; albeit with different matrices. The resultant sensitivities are appended to the [AMAT] matrix.

LINE 1021. This statement marks the end of the sensitivity boundary condition loop of the optimization phase of the standard MAPOL sequence. This loop began at Line 782.

LINE 1028. Calls the DESIGN module to perform automated redesign based on the constraint sensitivities in matrix [AMAT], the current active constraints in the CONST relation and the objective function gradient and the design variables in the GLBDES relation. The updated design variables are placed in GLBDES upon completion of the module.

LINE 1033. This statement marks the end of the design convergence loop that started at Line 238.

LINE 1034. This statement marks the end of the optimization phase of the standard sequence that started at Line 229.

#### C.8 FINAL ANALYSIS PHASE

LINE 1040. This IF test checks the existence of any analysis boundary conditions in the Solution Control. If none are present, control jumps to the end of the standard sequence at Line 1532.

LINE 1044. Initializes the constant "-1" used in the analysis phase.

LINE 1045. Calls the global matrix assembly module, EMA2. This module forms the stiffness and mass matrices in the structural set for the current design. Note that the resultant matrices always represent the model defined by the current design variable values.

#### C.9 ANALYSIS PHASE BOUNDARY CONDITION LOOP

LINE 1046. Initiates the analysis boundary condition loop for the ANALYSIS phase. After all analysis boundary conditions have been completed, execution will pass to termination at Line 1532. The loop terminates at Line 1531.

LINE 1047 TO 1050. Call the BOUND and BDCASE modules which determine the nature of the current boundary condition. Parameters are computed and passed to the MAPOL sequence defining the disciplines and matrix reductions to be performed. The parameters have the following definitions:

NMPC, NSPC, NOMIT, NRSET NGDR	The number of dependent degrees of freedom in each structural set. Set to -1 by BOUND if dynamic reduction is used in the current boundary condition.
BLOAD	Set to 1 if there are any static loads in the boundary condition.
BMASS	Set to 1 if the mass matrix is required by any discipline in the boundary condition.
BMODES	Set to 1 if any discipline in the boundary condition requires a normal modes analysis.
BSAERO	Set to 1 if there are any static aeroelastic analyses in the current boundary condition.
QDP	Dynamic pressure of the SAERO subcase(s).
MINDEX	Index for the appropriate steady aerodynamic matrices.
SYM	Symmetry flag for SAERO; - 1 if symmetric - 2 if antisymmetric.
TRMTYP	Type of trim for any SAERO disciplines; - 0 if no trim - 1 if lift only trim - 2 if lift/pitching moment trim.
BFLUTR	Set to 1 if any flutter analyses in the current boundary condition.
BDYN	Set to 1 if any dynamic response disciplines (except BLAST).
BDTR, BMTR	Set to 1 if any (D)irect or (M)odal (TR)ansient response analyses, respectively.
BDFR, BMFR	Set to 1 if any (D)irect or (M)odal (FR)equency response analyses, respectively.
BGUST	Set to 1 if any GUST options are selected in the current boundary condition.
BBLAST	Set to 1 if any BLAST analyses are in the current boundary condition.

Note that all subsequent operations within the analysis boundary condition loop will be directed by these flags.

LINE 1051. Calls the GTLOAD module to obtain the applied static loads representing the current design from the design load sensitivities and simple loads formed in LODGEN.

LINE 1052 TO 1060. Equivalence the proper aerodynamic influence coefficient matrix for steady aeroelastic analyses to the generic [AIC] matrix.

LINE 1064 TO 1090. This program block performs the required matrix reductions for Multipoint Constraints for all matrices that must be reduced, as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KGG], mass matrix [MGG], the static loads matrix [PG], SAERO splining

matrices [GTKG] and [GSTKG], and the unsteady aerodynamic splining matrix [UGTKG]. If no reductions are required, the n-set matrices are equivalenced to the g-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 1093 TO 1118. This program block performs the required matrix reductions for Single Point Constraints for all matrices that must be reduced, as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KNN], mass matrix [MNN], the static loads matrix [PN], SAERO splining matrices [GTKN] and [GSTKN], and the unsteady aerodynamic splining matrix [UGTKN]. If no reductions are required, the f-set matrices are equivalenced to the n-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 1122 TO 1127. This program block transforms the steady aerodynamic influence coefficient matrix [AIC] and the air loads matrix [AIRFRC] to the structural nodes using the SAERO spline matrices. The stiffness matrix [KAFF] is then formed which includes the aeroelastic stiffness based on [AICS] and QDP.

LINES 1129 TO 1226. This program block performs the required matrix reductions to the analysis set for all matrices that must be reduced as directed by the BOUND and BDCASE flags. These can include the stiffness matrix [KFF], mass matrix [MFF], the static loads matrix [PF], and the unsteady aerodynamic splining matrix [UGTKF]. There are three possible paths through this block of MAPOL code:

GENERAL DYNAMIC REDUCTION (LINES 1130 TO 1166)

The modules GDR1, GDR2, GDR3, and GDR4 are called to compute the necessary transformation matrix [GSUBO]. This matrix is then used to perform the matrix reductions.

STATIC REDUCTION (LINES 1168 TO 1212)

There are two subpaths through this block depending on whether the stiffness matrix is symmetric or asymmetric (which is the case for SAERO disciplines).

#### ASYMMETRIC STATIC REDUCTION (LINES 1172 TO 1192)

This block of code is used only for SAERO disciplines and assumes that there are support degrees of freedom (as is required for SAERO analyses). Lines 1177 to 1178 perform some operations on the symmetric stiffness matrix to ensure that the quantities needed to compute the [D] matrix for later support set reduction are available. Lines 1180 to 1182 reduce the asymmetric stiffness matrix to the analysis set. Note that the off diagonal partition [KOA], the asymmetric decomposed stiffness matrices [KOOU] and [KOOL] and the air loads on the omitted degrees of freedom [POARO] are output from FREDUCE for use in the recovery and design sensitivity operations. Lines 1186 to 1192 reduce the mass matrix to the analysis set. Note that the IFM matrix is computed for use in the support set reduction.

#### SYMMETRIC STATIC REDUCTION (LINES 1194 TO 1212)

This block of code is used for all disciplines other than SAERO using static reduction. Lines 1194 to 1195 reduce the symmetric stiffness matrix to the analysis set. Note that the symmetric decomposed stiffness matrices [KOOINV] and the static loads on the omitted degrees of freedom [PO] are output from FREDUCE for use in the recovery operations. Lines 1199 to 1205 reduce the mass matrix to the analysis set. Note that if there are also support degrees of freedom, the IFM matrix is computed for use in the support set reduction and sensitivity operations.

Lines 1207 to 1212 reduce the unsteady aerodynamic spline matrices to the analysis set. The roundabout order of operations is dictated by efficiency considerations for the MPYAD operation.

#### NO REDUCTION (LINES 1217 TO 1224)

If no reductions are required, the a-set matrices are equivalenced to the f-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 1227 TO 1309. This program block performs four functions: (1) support set reduction of the analysis set matrices, (2) static solution

(either SAERO or STATICS), (3) normal modes analysis, and (4) recovery of displacements and accelerations to the analysis set.

#### SUPPORT SET REDUCTION (LINES 1231 TO 1291)

Lines 1231 to 1233 compute the design invariant [D] matrix. Lines 1237 to 1242 compute the reduced mass matrix [MRR] and other matrices needed for inertia relief computations: [IFR], [R22] and [R21].

Lines 1247 to 1268 perform the steady aeroelastic analyses in the current boundary condition. Lines 1261 to 1262 call the SAERO module which performs the trim calculation. The matrix [DELTA] contains the vector of trim parameters on output. Note that the trimmed omitted loads [PO] are computed from the [POARO] and [DELTA] matrices. The displacements and accelerations are recovered to the analysis set in Lines 1264 to 1266 if there are displacements and accelerations computed. Zero degree of freedom aeroelastic analyses will not produce any displacement or acceleration fields. Note that this is different in the optimization phase where aeroelastic effectiveness constraints generate pseudo displacements for all trim type values.

Lines 1275 to 1288 perform static analysis with inertia relief. The INERTIA module is called on Line 1284 to compute the accelerations on the support points [AR]. The displacements and accelerations are then recovered to the analysis set in Lines 1285 to 1288.

Lines 1290 to 1291 perform real eigenvalue extraction including rigid body modes. The eigenvectors are output in the [PHIA] matrix and the LAMBDA relation contains the eigenvalue extraction data. Also output are the generalized mass matrix, [MII], and the number of extracted eigenvectors, HSIZE.

#### NO SUPPORT SET REDUCTION (LINES 1296 TO 1308)

In this case, all the computations are performed in the analysis set. Lines 1297 to 1298 solve for displacements due to static applied loads. Lines 1300 to 1306 represent a block of code needed for the currently non-functional capability to perform static aeroelastic analyses with input trim parameters. Lines 1307 to 1308 perform real eigenvalue extraction without rigid body modes and frequency con-

straint evaluation, respectively. The eigenvectors are output in the [PHIA] matrix and the LAMBDA relation contains the eigenvalue extraction data. Also output are the generalized mass matrix, [MII], and the number of extracted eigenvectors, HSIZE.

LINES 1314 TO 1332. This program block performs the requested dynamic response disciplines except for BLAST. Lines 1315 to 1316 call the QHHLGEN module to perform the modal reduction of the unsteady aerodynamic matrices for FLUTTER and GUST analyses to the dynamic set (including extra point degrees of freedom). Outputs are the [PHIKH] transformation matrix of normal modes at aerodynamic boxes, the [QHHL] matrix list for flutter and the [QHJL] matrix list for gust.

Lines 1317 to 1321 generate the stiffness, mass and damping matrices in the dynamic set (including extra point degrees of freedom) from DMA and the applied loads at time or frequency steps from DYNLOAD. [PDT] contains the direct or modal applied loads at time steps and [PDF] the direct or modal applied loads at frequency steps.

Lines 1322 to 1323 performs modal flutter analysis using the [KHFF] stiffness matrix, and the [MHH] generalized mass matrix from DMA and [QHHL] from QHHLGEN.

Lines 1325 to 1328 perform any TRANSIENT or FREQUENCY response analyses using the matrices generated in DMA and DYNLOAD. The outputs are either modal, [UxxxxI], or physical, [UxxxxA], and extra point [UxxxxE] displacements, velocities and accelerations. If modal analyses were performed, the physical set responses are computed in Lines 1329 to 1330.

LINES 1333 TO 1360. This program block performs BLAST analyses. The BLASTFIT module converts the unsteady aerodynamics matrices from the frequency domain to the time domain and also provides a number of matrices that are used in setting up a trim analysis. Lines 1337 through 1339 replace the rigid body modes calculated by REIG with the physical modes contained in the [D] matrix and Lines 1340 through 1354 set up generalized matrices (i.e., mass, stiffness, and aerodynamic). The BLASTTRIM module at Line 1355 performs a trim analysis prior to the blast encounter to set up initial conditions. The BLASTDRV module then calculates the aircraft response as a function of time. The outputs are the modal displacements, velocities and accelerations at the requested time steps in the matrix [UBLASTI].

LINES 1364 TO 1426. This program block performs the required matrix recovery from the analysis set (a-set) to the free degrees of freedom (f-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the static displacement matrix [UF], the acceleration matrix [AF], the eigenvectors [PHIF] and the transient and frequency physical response matrices [UTRANF] and [UFREQF]. There are three possible paths through this block of MAPOL code:

RECOVERY WITH GENERAL DYNAMIC REDUCTION (LINES 1369 TO 1379)

The [GSUBO] transformation matrix computed during matrix reduction is used to recover the response quantities at the free degrees of freedom.

RECOVERY WITH STATIC REDUCTION (LINES 1381 TO 1409)

Like the corresponding reduction, there are two recovery paths depending on the symmetry of the stiffness matrix. Lines 1386 to 1389 recover the displacements and accelerations using the symmetric [KOOINV] and [GSUBO] matrices computed in the symmetric reduction path. Lines 1392 to 1396 recover the displacements and accelerations using the asymmetric decomposed omitted stiffness partition [KOOU] and [KOOL] and [GASUBO] computed in the asymmetric path. Lines 1398 to 1409 perform recovery operations for normal modes, transient response and frequency response disciplines, all of which require that symmetric stiffness reduction was performed.

RECOVERY WITH NO REDUCTIONS (LINES 1414 TO 1425)

If no reductions were applied, the f-set matrices are equivalenced to the a-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINES 1428 TO 1464. This program block performs the required matrix recovery from the free degrees of freedom (f-set) to the independent set (n-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the statics displacement matrix [UN], the acceleration matrix [AN], the eigenvectors [PHIN] and the transient and frequency physical response matrices [UTRANN] and [UFREQN]. If no reductions were applied, the n-set matrices are equivalenced to the f-set matrices. The



program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINE 1466 TO 1514. This program block performs the required matrix recovery from the independent set (n-set) to the global structural set (g-set) for all matrices that must be recovered as directed by the BOUND and BDCASE flags. These can include the statics displacement matrix [UG], the acceleration matrix [AG], the eigenvectors [PHIG] and the transient and frequency physical response matrices [UTRANG] and [UFREQG]. If no reductions were applied, the g-set matrices are equivalenced to the n-set matrices. The program block begins with a NULLMAT call which breaks matrix equivalences set in previous boundary conditions.

LINE 1518. The modal blast responses are recovered to the physical set.

LINE 1522 TO 1529. This program block handles output requests not handled directly within the analysis modules. The OFPLOAD module satisfies applied load output requests and the OFPDISP module satisfies displacement, velocity, acceleration and real eigenvalue (ROOTS) output requests. The EDR module computes requested element response quantities (e.g., stress and strain) and the computed quantities are printed in OFPEDR.

LINE 1530. This statement marks the end of the analysis boundary condition loop of the analysis phase of the standard MAPOL sequence.

LINE 1531. This statement marks the end of the analysis phase of the standard sequence that started at Line 1040.

LINE 1532. Termination of the standard MAPOL sequence.

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
1 11$*****$!
2 11$ EXECUTIVE SEQUENCE FOR ASTROS — 11 FEBRUARY 1988 $!
3 11$ $!
4 11$*****$!
5 11$ VARIABLE DECLARATION SEGMENT $!
6 11$*****$!
7 11$ $!
8 11INTEGER NEGL; $!
9 11INTEGER GSIZE, CONVERGE, NITER, MAXITER, MAXFSD, $!
10 11 FSDFLG, BC, NDV; $!
11 11REAL CTL, CTLMIN, MOVLIM, CNVRGLIM, ALPHA; $!
12 11UNSTRUCT DCENT, GRIDTEMP; $!
13 11RELATION DESHIST, CONST, MPPARM, CONVERT, SAVE, $!
14 11 MFORM, GRID, SPOINT, EPOINT, SEQGP, $!
15 11 BGPDT, CSTM, FORCE, FORCE1, MOMENT, $!
16 11 MOMENT1, PLOAD, GRAV, LOAD, EIGR, $!
17 11 TEMP, TEMPD, DCONSP, DCONFRQ, DCONTHK, $!
18 11 CORD1C, CORD1R, CORD1S, CORD2C, CORD2R, $!
19 11 CORD2S; $!
20 11$ $!
21 11$*****$!
22 11$ DECLARATIONS FOR MODULE MKUSET $!
23 11$*****$!
24 11$ $!
25 11UNSTRUCT USET; $!
26 11RELATION SPC, SPC1, SPCADD, MPC, MPCADD, $!
27 11 ASET, ASET1, OMIT, OMIT1, SUPORT, $!
28 11 JSET, JSET1; $!
29 11MATRIX [PGMN(8)], [PNSF(8)], [PFOA(8)], [PARL(8)], [TMN(8)], $!
30 11 [YS(8)]; $!
31 11$ $!
32 11$*****$!
33 11$ DECLARATIONS FOR MODULES MAKEST AND EMG $!
34 11$*****$!
35 11$ $!
36 11UNSTRUCT DVSIZE, VFIXD, PCOMPS; $!
37 11UNSTRUCT KELM, MELM, TELM; $!
38 11RELATION QDMM1, QDMM1EST, CROD, CONROD, RODEST, $!
39 11 CSHEAR, SHEAREST, CTRMEM, TRMEMEST, CMAS1, $!
40 11 CMAS2, MASSEST, CONM1, CONM1EST, CONM2, $!
41 11 CONM2EST, CBAR, BEAMEST, CQUAD4, QUAD4EST, $!
42 11 CIHEX1, IHEX1EST, CIHEX2, IHEX2EST, CIHEX3, $!
43 11 IHEX3EST, CELAS1, CELAS2, ELASEST, ELIST, $!
44 11 PLIST, DVCT, DESVAR, GLBDES, DESELM, $!
45 11 LOCLVAR, PCOMP, PQDMM1, PROD, PSHEAR, $!
46 11 PTRMEM, PMASS, PELAS, PBAR, PSHELL, $!
47 11 PCOMP1, PCOMP2, PIHEX, MAT1, MAT2, $!
48 11 MAT8, MAT9, DCONSTR; $!
49 11MATRIX [PTRANS]; $!
50 11MATRIX [PMINT], [PMAXT], [SMAT]; $!
51 11$ $!
52 11$*****$!
53 11$ DECLARATIONS FOR ELEMENT DATA RECOVERY (EDR) $!
54 11$*****$!
55 11$ $!
56 11RELATION GRIDLIST, MODELIST, ELEMLIST, FREQLIST, TIMELIST, $!
57 11 GPFELEM, EOSUMTRY, EOBAR, EOELAS, EOHEX1, $!
58 11 EOHEX2, EOHEX3, EOQDMM1, EOQUAD4, EOROD, $!
59 11 EOSHEAR, EOTRMEM, GPFDATA; $!
60 11UNSTRUCT EODISC; $!

```

Figure C.1 Standard MAPOL Sequence

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
61 11$
62 11$*****$!
63 11$      DECLARATIONS FOR MODULES EMA1, EMA2 AND GLOBAL      $!
64 11$      MATRIX PARTITION/REDUCTION                          $!
65 11$*****$!
66 11$
67 11UNSTRUCT  DKVI,      DMVI;
68 11RELATION  GMKCT,      GMMCT;
69 11MATRIX    ]KGG],      ]KNN],      ]KFF],      ]KAA],      ]KLL],
70 11          ]MGG],      ]MNN],      ]MFF],      ]MAA],      ]MLL],
71 11          ]MRRBAR],    ]MLR],      ]KFS],      ]KSS],      ]KOOINV(8)],
72 11          ]GSUBO(8)],  ]KOOL(8)],  ]KOOU(8)],  ]KLLINV(8)],  ]KAO(8)],
73 11          ]MRR(8)],    ]IFM(8)],  ]IFR(8)],  ]D(8)],      ]KLR],
74 11          ]LHS(8)],    ]KLU(8)],  ]KLL(8)],  ]MOO],      ]MOA],
75 11          ]MAABAR];
76 11MATRIX    ]TMP1],      ]TMP2];
77 11MATRIX    ]PG],        ]PN],        ]PF],        ]PA],        ]PL],
78 11          ]PO],        ]PLBAR],    ]PR],        ]RHS(8)],  ]UG(8)],
79 11          ]UN],        ]UF],        ]UA],        ]UL],        ]UM],
80 11          ]AG(8)],      ]AN],        ]AF],        ]AA],        ]AR],
81 11          ]AL],        ]UO],        ]UOO],      ]POARO(8)];
82 11$
83 11$*****$!
84 11$      DECLARATIONS FOR SOLUTION CONTROL                    $!
85 11$*****$!
86 11$
87 11INTEGER    NUMOPTBC,    OPSTRAT,    NBNDCOND,    BLOAD,      BMASS,
88 11          BMODES,      BSAERO,    BFLUTR,    BDYN,      BDRSP,
89 11          BDTR,        BMTR,      BDFR,      BMFR,      BGUST,
90 11          BBLAST,      NMPC,      NSPC,      NOMI,      NRSET,
91 11          TRMTYP;
92 11RELATION    CASE;
93 11$
94 11$*****$!
95 11$      DECLARATIONS FOR SENSITIVITY EVALUATION              $!
96 11$*****$!
97 11$
98 11INTEGER    DDFLG,      NACSD,      AAC,      NAU,      ABC,
99 11          ADC,        AFC,      NAE,      NAUE;
100 11REAL      EPS,        NRFAC;
101 11UNSTRUCT  PCA,        PAE;
102 11MATRIX    ]DFDU],      ]PGA],      ]UGA],      ]DUG],      ]DMUG],
103 11          ]DPFV],      ]DPOV],      ]DPNV],      ]DPAV],      ]DUAV],
104 11          ]DUAD],      ]DUFV],      ]AGA],      ]AMAT],      ]DKUG],
105 11          ]DPGV],      ]DPLV],      ]DURD],      ]DULD],      ]DULV],
106 11          ]DDELDV],    ]DPRV],      ]DKLV],      ]DRHS],      ]DFDUF],
107 11          ]DFDUN],      ]DMAG],      ]DMUN],      ]DMUF],      ]DMUA],
108 11          ]DMUO],      ]DMUL],      ]DMUR],      ]DMU],      ]DP1],
109 11          ]DKIV];
110 11MATRIX    ]GLBSIG],    ]DPTHVI],  ]DPGRVI],  ]DPVJ];
111 11$
112 11$*****$!
113 11$      AERODYNAMIC ENTITIES                                $!
114 11$*****$!
115 11$
116 11INTEGER    SYM,        MINDEX,    NAERO;
117 11REAL      QDP;
118 11UNSTRUCT  ACPT,      UNMK;

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
119 11RELATION AESURF, AIRFOIL, AEROS, AEFAC, AXSTA,
120 11 BODY, SPLINE1, SET1, SET2, ATTACH,
121 11 TRIM, AERO, BLAST, CAERO6, PAERO6,
122 11 USAGEOM, AECOMP, STABCF, CAERO1, PAERO1,
123 11 CAERO2, PAERO2, MKAERO1, MKAERO2, FLUTTER,
124 11 FLFACT, CLAMBDA, DCONALE, DCONCLA, DCONFLT;
125 11MATRIX [AIRFC(8)], [AICMAT(8)], [AAICMAT(8)], [AIC], [AICS],
126 11 [KAFF], [PAF], [KAAA], [PAA], [GASUBO(8)],
127 11 [SKJ], [DIJK], [D2JK], [KARL], [KALL],
128 11 [K21(8)], [PARBAR], [PAL], [PAR(8)], [K112(8)],
129 11 [DELTA], [GTKG], [GTKN], [GTKF], [GSTKG],
130 11 [GSTKN], [GSTKF], [GSKF], [UGTKG], [UGTKN],
131 11 [UGTKF], [UGTKA], [UGTKO], [UGTKAB], [AITD],
132 11 [KARR], [KALR], [R22], [R32], [K11],
133 11 [K12(8)], [P1], [R21], [R31], [KL11(8)],
134 11 [KU11(8)], [P2];
135 11MATRIX [AJJTL], [QJL], [QKKL], [QHHL(8)];
136 11$
137 11$*****$!
138 11$ DYNAMIC RESPONSE DECLARATIONS $!
139 11$*****$!
140 11$ $!
141 11INTEGER HSIZE;
142 11UNSTRUCT TFDATA, ICDATA;
143 11RELATION LAMBDA, OEIGS, DONLY, DLOAD, TABLED1,
144 11 IC, TLOAD1, TLOAD2, RLOAD1, RLOAD2,
145 11 TSTEP, VSDAMP, TABDMP1, DLAGS, TF,
146 11 DMIG, GUST, FREQ, FREQ1, FREQ2,
147 11 FFT, FLUTREL;
148 11MATRIX [PHIK], [QHJL], [QKJL], [PHIA], [MI],
149 11 [PHIO], [PHIF], [PHIN], [PHIG(8)], [KHHT],
150 11 [KHIF(8)], [BHH], [MHH(8)], [PDT], [PDF],
151 11 [KDDT], [KDDF], [BDD], [MDD], [IMATRIX],
152 11 [UTRANA], [UFREQA], [UTRANI], [UFREQI], [UFREQE],
153 11 [UTRANE], [UTRANF], [UFREQF], [UTRANN], [UFREQN],
154 11 [UTRANG], [UFREQG], [FLUTMODE], [PTGLOAD], [PFGLOAD];
155 11$ $!
156 11$*****$!
157 11$ DECLARATIONS FOR GENERAL DYNAMIC REDUCTION (GDR) $!
158 11$*****$!
159 11$ $!
160 11INTEGER LKSET, LJSET, NEIV, GNORM, NGDR,
161 11 GSIZE;
162 11REAL FMAX;
163 11UNSTRUCT GDRUSET;
164 11RELATION DYNRED, GDRBGPD;
165 11MATRIX [GDRGO(8)], [PHIOK], [KOO], [GGO], [KSOO],
166 11 [KOA], [LSOO];
167 11$ $!
168 11$*****$!
169 11$ BLAST RESPONSE DECLARATIONS $!
170 11$*****$!
171 11$ $!
172 11REAL BQDP;
173 11MATRIX [MPART], [ID2], [PHIE], [PHIR], [PHIB],
174 11 [GENM], [GENK], [GENF], [GENQ], [GENQL],
175 11 [DTSLP], [FTF], [QRE], [QEE], [KEQE],
176 11 [LKQ], [UKQ], [GFR], [GFE], [BTEM],
177 11 [BLSTJA], [BLGTJA], [BFR], [MATTR], [MATSS],
178 11 [KEE], [DELB], [DELM], [URDB], [GENFA],
179 11 [DMNWSH], [ELAS], [SLPMOD], [QRR], [UBLASTI],
180 11 [UBLASTG];

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
181 11$
182 11$*****$!
183 11$
184 11$          BEGIN MAPOL SOLUTION SEQUENCE
185 11$
186 11$*****$!
187 11$          PREFACE MODULES
188 11$*****$!
189 11$
190 11CALL SOLUTION( NUMOPTBC , NBNDCOND, OPSTRAT );
191 11CALL IFP ( GSIZE );
192 11FOR BC = 1 TO NBNDCOND DO
193 21  CALL MKUSET( BC, GSIZE, {YS(BC)}, {TMN(BC)}, {PGMN(BC)}, {PNSF(BC)},
194 21      {PFOA(BC)}, {PARL(BC)});
195 21ENDDO;
196 11$
197 11$          GENERATE THE ELEMENT MATRICES
198 11$
199 11CALL MAKEST ( NDV );
200 11CALL EMG ( NDV, GSIZE );
201 11CALL PFBULK ( GSIZE, EOSUMTRY, EODISC, GPFELEM );
202 11$
203 11$          ASSEMBLE THE ELEMENT MATRICES
204 11$          TO THE SENSITIVITY MATRICES
205 11$
206 11CALL EMA1 ( NDV, GMKCT, DKVI, GMMCT, DMVI );
207 11$
208 11$          GENERATE THE SIMPLE LOAD VECTORS
209 11$          AND LOAD SENSITIVITIES
210 11$
211 11CALL LODGEN ( GSIZE );
212 11$
213 11$          GENERATE THE AIC MATRIX AND THE
214 11$          SPLINE TRANSFORMATION MATRICES
215 11$
216 11MINDEX := 1;
217 11NAERO := 2;
218 11WHILE MINDEX <= NAERO DO
219 21  CALL PFAERO ( GSIZE, {AICMAT(MINDEX)}, {AAICMAT(MINDEX)},
220 21      {AIRFR(MINDEX)}, MINDEX, NAERO,
221 21      {GTKG}, {GSTKG}, {UGTKG}, {AJJTL}, {D1JK}, {D2JK}, {SKJ} );
222 21ENDDO;
223 11CALL AMP ( {AJJTL}, {D1JK}, {D2JK}, {SKJ}, {QKKL}, {QKJL}, {QJJL} );
224 11$
225 11$*****$!
226 11$          BEGIN OPTIMIZATION LOOP
227 11$*****$!
228 11$
229 11IF NUMOPTBC > 0 THEN
230 21  CONVERGE := 0;
231 21  NITER := 0;
232 21  MAXITER := 15;
233 21  MOVLIM := 2.0;
234 21  MAXFSD := 3;
235 21  ALPHA := 0.90;
236 21  CNVRGLIM := 0.50;
237 21  NEG1 := -1;
238 21  WHILE CONVERGE < 2 AND NITER <= MAXITER DO
239 31$

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
240 31$ ASSEMBLE THE GLOBAL MATRICES $!
241 31$ $!
242 31 NITER := NITER + 1; !
243 31 CALL TCEVAL ( MOVLIM ); !
244 31 CALL EMA2 ( GSIZE, [KGG], [MGG], NITER ); !
245 31$ $!
246 31$ BEGIN BOUNDARY CONDITION LOOP FOR OPTIMIZATION $!
247 31$ $!
248 31 FOR BC = 1 TO NUMOPTBC DO !
249 41 CALL BOUND (BC, NMPC, NSPC, NOMIT, NRSET, NGDR ); !
250 41 CALL BDCASE(BC, BLOAD, BMAS, BMODES, BSAERO, QDP, MINDEX, !
251 41 SYM, TRMTYP, BFLUTR, BDRSP, BDRSP, BDRSP, BDRSP, BDRSP, BDRSP, !
252 41 BGUST, BBLAST ); !
253 41 IF BLOAD <> 0 CALL GTLOAD ( BC, GSIZE, [PG] ); !
254 41 IF BSAERO <> 0 THEN !
255 51 CALL NULLMAT ( [AIC] ); !
256 51 IF SYM = 1 THEN !
257 61 [AIC] := [AICMAT(MINDEX)]; !
258 61 ELSE IF SYM = -1 THEN !
259 71 [AIC] := [AAICMAT(MINDEX)]; !
260 71 ENDIF; !
261 61 ENDIF; !
262 51 ENDIF; !
263 41$ $!
264 41$ PARTITION-REDUCTION OF GLOBAL MATRICES $!
265 41$ $!
266 41 IF NUMOPTBC > 1 CALL NULLMAT ( [KNN], [PN], [MNN], !
267 51 [GTKN], [GSTKN], [UGTKN] ); !
268 41 IF NMPC <> 0 THEN !
269 51$ $!
270 51$ PERFORM MPC REDUCTION $!
271 51$ $!
272 51 CALL GREduce ( [KGG], [PG], [PGMN(BC)], [TMN(BC)], [KNN], [PN] ); !
273 51 IF BMAS <> 0 CALL GREduce ( [MGG], [PGMN(BC)], [TMN(BC)], [MNN] ); !
274 51 IF BSAERO <> 0 THEN !
275 61 CALL GREduce (, [GTKG], [PGMN(BC)], [TMN(BC)],, [GTKN]); !
276 61 CALL GREduce (, [GSTKG], [PGMN(BC)], [TMN(BC)],, [GSTKN]); !
277 61 ENDIF; !
278 51 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 !
279 61 CALL GREduce (, [UGTKG], [PGMN(BC)], [TMN(BC)],, [UGTKN] ); !
280 51 ELSE !
281 51$ $!
282 51$ NO MPC REDUCTION $!
283 51$ $!
284 51 [KNN] := [KGG]; !
285 51 IF BLOAD <> 0 [PN] := [PG]; !
286 51 IF BMAS <> 0 [MNN] := [MGG]; !
287 51 IF BSAERO <> 0 THEN !
288 61 [GTKN] := [GTKG]; !
289 61 [GSTKN] := [GSTKG]; !
290 61 ENDIF; !
291 51 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKN] := [UGTKG]; !
292 51 ENDIF; !
293 41 IF NUMOPTBC > 1 CALL NULLMAT ( [KFF], [PF], [MFF], [GTKF], [GSTKF], !
294 51 [UGTKF] ); !
295 41 IF NSPC <> 0 THEN !
296 51$ $!
297 51$ PERFORM SPC REDUCTION $!
298 51$ $!
299 51 CALL NREDUCE ( [KNN], [PN], [PNSF(BC)], [YS(BC)], [KFF], [KFS], !
300 51 [KSS], [PF]); !

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT  LEVEL
301      51      IF BMASS <> 0 CALL NREDUCE ( [MNN], , [PNSF(BC)], , [MFF] );
302      51      IF BSAERO <> 0 THEN
303      61          CALL NREDUCE ( , [GTKN], [PNSF(BC)], , , , [GTKF] );
304      61          CALL NREDUCE ( , [GSTKN], [PNSF(BC)], , , , [GSTKF] );
305      61      ENDIF;
306      51      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0
307      61          CALL NREDUCE ( , [UGTKN], [PNSF(BC)], , , , [UGTKF] );
308      51      ELSE
309      51$
310      51$      NO SPC REDUCTION
311      51$
312      51          [KFF] := [KNN];
313      51      IF BLOAD <> 0 [PF] := [PN];
314      51      IF BMASS <> 0 [MFF] := [MNN];
315      51      IF BSAERO <> 0 THEN
316      61          [GTKF] := [GTKN];
317      61          [GSTKF] := [GSTKN];
318      61      ENDIF;
319      51      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKF] := [UGTKN];
320      51      ENDIF;
321      41$
322      41$      IF STEADY AERO FOR THIS B.C., ADJUST THE KFF MATRIX
323      41$
324      41      IF BSAERO <> 0 THEN
325      51          IF NITER = 1 OR NUMOPTBC > 1 THEN
326      61              CALL TRNPOSE ( [GSTKF], [GSKF] );
327      61              [AICS] := [GTKF] * [ TRANS([AIC]) * [GSKF] ];
328      61              [PAF] := (QDP) ( [GTKF] * [AIRFR(C(MINDEX)) ] );
329      61          ENDIF;
330      51          [KAFF] := [KFF] - (QDP) [AICS];
331      51      ENDIF;
332      41$
333      41      IF NUMOPTBC > 1 CALL NULLMAT ( [KAA], [PA], [MAA],
334      51          [KAAA], [PAA], [UGTKA] );
335      41
336      51$      IF NGDR <> 0 THEN
337      51$      PERFORM THE GENERAL DYNAMIC REDUCTION
338      51$
339      51      IF NOMIT <> 0 OR NRSET <> 0 THEN
340      61$
341      61$      OBTAIN THE OMITTED DOF PARTITION OF KFF AND MFF
342      61$
343      61          CALL PARTN ( [KFF], [KOO], , [KOA], , [PFOA(BC)] );
344      61          CALL PARTN ( [MFF], [MOO], , , , [PFOA(BC)] );
345      61      ELSE
346      61          [KOO] := [KFF];
347      61          [MOO] := [MFF];
348      61      ENDIF;
349      51      CALL GDRI ( [KOO], [MOO], [KSOO], [GGO], LKSET, LJSET, NEIV,
350      51          FMAX, BC );
351      51$
352      51$      LKSET      MEANING
353      51$      <> 0      K-SET EXISTS
354      51$      = 0      K-SET DOES NOT EXIST
355      51$
356      51      IF LKSET <> 0 THEN
357      61          CALL SDCOMP ( [KSOO], [LSOO] );
358      61          CALL GDR2 ( [LSOO], [MOO], [PHIOK], LKSET, LJSET,
359      61              NEIV, FMAX, BC );
360      61      ENDIF;

```

Figure G.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
361 51      CALL GDR3 ( [KOO], [KOA], [MGG], [PHIOK], [TMN(BC)], [GGO],
362 51      [PGMN(BC)], [FNSF(BC)], [PFOA(BC)], [GSUBO(BC)],
363 51      LKSET, LJSET, NOMIT, NRSET, GNORM, BC );
364 51      CALL GDR4 ( BC, GSIZE, GSIZE, LKSET, LJSET, NUMOPTBC,
365 51      NBNDCOND, [GDRGO(BC)], , , , [PARL(BC)] );
366 51      [MAA] := TRANS ( [GSUBO(BC)] ) * [ [MFF] * [GSUBO(BC)] ];
367 51      [KAA] := TRANS ( [GSUBO(BC)] ) * [ [KFF] * [GSUBO(BC)] ];
368 51      IF BLOAD <> 0 [PA] := TRANS ( [GSUBO(BC)] ) * [PF];
369 51      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
370 61      [TMP1] := TRANS ( [UGTKF] ) * [GSUBO(BC)];
371 61      CALL TRNSPOSE ( [TMP1], [UGTKA] );
372 61      ENDIF;
373 51      ELSE
374 51      IF NOMIT <> 0 THEN
375 61$
376 61$      PERFORM THE STATIC REDUCTION
377 61$
378 61      IF BSAERO <> 0 THEN
379 71      IF NITER = 1 AND BLOAD = 0 THEN
380 81$
381 81$      FORM [KAA] ON FIRST PASS SO [D] CAN BE FORMED
382 81$
383 81      CALL FREDUCE ( [KFF], , [PFOA(BC)], , [KOOINV(BC)], , ,
384 81      [GSUBO(BC)], [KAA] );
385 81      ENDIF;
386 71      CALL FREDUCE ( [KAPF], [PAF], [PFOA(BC)], BSAERO,
387 71      [KOOL(BC)], [KOOU(BC)], [KAO(BC)],
388 71      [GASUBO(BC)], [KAAA], [PAA], [POARO(BC)] );
389 71$
390 71$      PERFORM GUYAN REDUCTION OF THE MASS MATRIX
391 71$
392 71      IF BMASS <> 0 THEN
393 81      CALL PARTN ( [MFF], [MOO], , [MOA], [MAABAR], [PFOA(BC)] );
394 81      [MAA] := [MAABAR] + TRANS([MOA]) * [GASUBO(BC)] +
395 81      TRANS([GASUBO(BC)]) * [MOA] +
396 81      TRANS([GASUBO(BC)]) * [ [MOO] * [GASUBO(BC)] ];
397 81      IF NRSET <> 0 [IFM(BC)] := [MOO] * [GASUBO(BC)] + [MOA];
398 81      ENDIF;
399 71      ELSE
400 71      CALL FREDUCE ( [KFF], [PF], [PFOA(BC)], , [KOOINV(BC)], , ,
401 71      [GSUBO(BC)], [KAA], [PA], [PO] );
402 71$
403 71$      PERFORM GUYAN REDUCTION OF THE MASS MATRIX
404 71$
405 71      IF BMASS <> 0 THEN
406 81      CALL PARTN ( [MFF], [MOO], , [MOA], [MAABAR], [PFOA(BC)] );
407 81      [MAA] := [MAABAR] + TRANS([MOA]) * [GASUBO(BC)] +
408 81      TRANS([GASUBO(BC)]) * [MOA] +
409 81      TRANS([GASUBO(BC)]) * [ [MOO] * [GASUBO(BC)] ];
410 81      IF NRSET <> 0 [IFM(BC)] := [MOO] * [GASUBO(BC)] + [MOA];
411 81      ENDIF;
412 71      ENDIF;
413 61      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
414 71      CALL ROWPART ( [UGTKF], [UGTKO], [UGTKAB], [PFOA(BC)] );
415 71      [TMP1] := TRANS ( [UGTKO] ) * [GSUBO(BC)];
416 71      CALL TRNSPOSE ( [TMP1], [TMP2] );
417 71      [UGTKA] := [UGTKAB] + [TMP2];
418 71      ENDIF;
419 61      ELSE

```

Figure C.1 Standard MAPOL Sequence (Continued)



\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
420 61$ NO F-SET REDUCTION $!
421 61$ $!
422 61$ $!
423 61 [KAA] := [KFF]; $!
424 61 IF BLOAD <> 0 [PA] := [PF]; $!
425 61 IF BSAERO <> 0 THEN $!
426 71 [KAAA] := [KAPF]; $!
427 71 [PAA] := [PAF]; $!
428 71 ENDIF; $!
429 61 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKA] := [UGTKF]; $!
430 61 IF BMASS <> 0 [MAA] := [MFF]; $!
431 61 ENDIF; $!
432 51 ENDIF; $!
433 41$ $!
434 41 IF NRSET <> 0 THEN $!
435 51$ $!
436 51$ $!
437 51$ $!
438 51 $!
439 61 $!
440 61 $!
441 61 $!
442 61 $!
443 61 $!
444 71 $!
445 71 $!
446 71 $!
447 61 $!
448 51$ $!
449 51$ $!
450 51$ $!
451 51 $!
452 51 $!
453 51 $!
454 51 $!
455 51 $!
456 51 $!
457 51 $!
458 61$ $!
459 61$ $!
460 61$ $!
461 61 $!
462 61 $!
463 61 $!
464 61 $!
465 61 $!
466 61 $!
467 61 $!
468 61 $!
469 61 $!
470 61 $!
471 61 $!
472 61 $!
473 61 $!
474 61 $!
475 61 $!
476 61 $!
477 61 $!
478 61 $!
479 61 $!
480 61 $!
481 61 $!
482 61 $!

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
483 51$
484 51
485 61$
486 61$
487 61$
488 61
489 61
490 61
491 61
492 61
493 61
494 61
495 61
496 61
497 61
498 61
499 61
500 61
501 61
502 61
503 51
504 61
505 61
506 61
507 61
508 51
509 51$
510 51$
511 51$
512 51
513 61
514 61
515 61
516 51
517 61$
518 61$
519 61$
520 61
521 61
522 61
523 51
524 61
525 61
526 61
527 51
528 41$
529 41$
530 41$
531 41$
532 41
533 51
534 51
535 51
536 51
537 51
538 51
539 51
540 51
541 61

IF BLOAD <> 0 THEN
    PROCESS STATICS WITH INERTIA RELIEF
    CALL MERGE ( [K11], [R22], [KLR], [R21],
                 [KLL], [PARL(BC)] );
    CALL YSMERGE ( [P1], , [PLBAR], [PARL(BC)] );
    CALL YSMERGE ( [K12(BC)], , [IFR(BC)], [PARL(BC)] );
    CALL DECOMP ( [K11], [KL11(BC)], [KU11(BC)] );
    [P1] := [P1] - [K12(BC)] * [AR];
    CALL GFBS ( [KL11(BC)], [KU11(BC)], [P1], [UA], NEG1 );
    [LHS(BC)] := [MRR(BC)];
    [RHS(BC)] := TRANS([D(BC)]) * [PLBAR] + [PR];
    CALL INERTIA ( [LHS(BC)], [RHS(BC)], [AR] );
    [AL] := [D(BC)] * [AR];
    CALL ROMERGE ( [AA], [AR], [AL], [PARL(BC)] );
    [P1] := [P1] - [K12(BC)] * [AR];
    CALL GFBS ( [KL11], [KU11], [P1], [UA] );
ENDIF;
IF BMODES <> 0 THEN
    CALL REIG ( BC, [KAA], [MAA], [MRR(BC)], [D(BC)], LAMBDA,
               [PHIA], [MII], HSIZE );
    CALL FCEVAL ( BC, LAMBDA );
ENDIF;
ELSE
    NO SUPPORT SET REDUCTION
    IF BLOAD <> 0 THEN
        CALL SDCOMP ( [KAA], [KLLINV(BC)] );
        CALL FBS ( [KLLINV(BC)], [PA], [UA] );
    ENDIF;
    IF BSAERO <> 0 THEN
        NOT FUNCTIONAL, REQUIRES [DELTA] TO BE INPUT
        CALL DECOMP ( [KAAA], [KLL(BC)], [KLU(BC)] );
        CALL GFBS ( [KLL(BC)], [KLU(BC)], [PAA], [UA] );
    ENDIF;
    IF BMODES <> 0 THEN
        CALL REIG ( BC, [KAA], [MAA], , LAMBDA, [PHIA], [MII], HSIZE );
        CALL FCEVAL ( BC, LAMBDA );
    ENDIF;
ENDIF;
PERFORM ANY DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT
OF THE SUPPORT SET
IF BDYN <> 0 THEN
    CALL QHNLGEN ( BC, [QKKL], [QKJL], [UGTKA], [PHIA],
                  [PHIKH], [QHHL(BC)], [QHJL] );
    CALL DMA ( BC, GSIZE, [MAA], [KAA], [TMN(BC)], [GSUBO(BC)], NGDR,
              LAMBDA, [PHIA], [MDD], [BDD], [KDDT], [KDDF],
              [MHH(BC)], [BHH], [KHHT], [KHHF(BC)] );
    CALL DYNLOAD ( BC, GSIZE, [TMN(BC)], [GSUBO(BC)], NGDR,
                  [PHIA], [QHJL], [PDT], [PDF] );
    IF BFLUTR <> 0 CALL FLUTTRAN ( BC, [QHHL(BC)], LAMBDA, HSIZE,
                                  [MHH(BC)], [KHHF(BC)] );

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
542 51      IF BDRSP <> 0 THEN
543 61          CALL DYNRSP ( BC, [MDD], [BDD], [KDDT], [KDDF], [MH(BC)], [BHH],
544 61              [KHHT], [KHHT(BC)], [PDT], [PDF], [QHHL(BC)],
545 61              [UTRANA], [UFREQA], [UTRANI], [UFREQI], [UTRANE],
546 61              [UFREQE] );
547 61          IF BMTR <> 0 [UTRANA] := [PHIA] * [UTRANI];
548 61          IF BMFR <> 0 [UFREQA] := [PHIA] * [UFREQI];
549 61      ENDIF;
550 51
551 41      IF BBLAST <> 0 THEN
552 51          CALL BLASTFIT ( BC, [QJL], [MATR], [MATSS], [BQDP], [BFRC],
553 51              [DWNWSH], HSIZE, [ID2], [MPART], [UGTKA],
554 51              [BLGTJA], [BLSTJA] );
555 51          CALL COLPART ( [PHIA], , [PHIE], [MPART] );
556 51          CALL ROWMERGE ( [PHIR], [ID2], [D(BC)], [PARL(BC)] );
557 51          CALL COLMERGE ( [PHIB], [PHIR], [PHIE], [MPART] );
558 51          [GENM] := TRANS ( [PHIB] ) * [ [MAA] * [PHIB] ];
559 51          [GENK] := TRANS ( [PHIB] ) * [ [KAA] * [PHIB] ];
560 51          [DTSJP] := TRANS ( [BLSTJA] ) * [PHIB];
561 51          [FTF] := TRANS ( [PHIB] ) * [BLGTJA];
562 51          [GENF] := [BQDP] [FTF] * [BFRC];
563 51          [GENFA] := [BQDP] [FTF] * [MATSS];
564 51          [GENQ] := [GENFA] * [DTSJP];
565 51          [GENQL] := [BQDP] [FTF] * [MATR];
566 51          CALL PARTN ( [GENQ], [QRR], , [QRE], [QEE], [MPART] );
567 51          CALL PARTN ( [GENK], , , [KEE], [MPART] );
568 51          [KEQE] := [QEE] + [KEE];
569 51          CALL DECOMP ( [KEQE], [LKQ], [UKQ] );
570 51          CALL ROWPART ( [GENF], [GFR], [GFE], [MPART] );
571 51          CALL GFBS ( [LKQ], [UKQ], [GFE], [BTEM] );
572 51          [DELM] := -[QRE] * [BTEM] + [GFR];
573 51          CALL BLASTRIM ( BC, [DELM], [MRR(BC)], [URDB], [DELB] );
574 51          [ELAS] := [BTEM] * [DELB];
575 51          [SLPMOD] := TRANS ( [BLSTJA] ) * [PHIE];
576 51          CALL BLASTDRV ( BC, [GENM], [GENK], [GENFA], [GENQL], [DELB],
577 51              [URDB], [DWNWSH], [SLPMOD], [ELAS], [UBLASTI] );
578 51      ENDIF;
579 41$
580 41$      BEGIN THE DATA RECOVERY OPERATIONS
581 41$
582 41      IF NUMOPTBC > 1 CALL NULLMAT ([UF], [AF], [PHIF], [UTRANF], [UFREQF] );
583 41      IF NGDR <> 0 THEN
584 51$
585 51$          DATA RECOVERY WITH GDR
586 51$
587 51          IF BLOAD <> 0 THEN
588 61              [UF] := [GSUBO(BC)] * [UA];
589 61              IF NRSET <> 0 [AF] := [GSUBO(BC)] * [AA];
590 61          ENDIF;
591 51          IF BSAERO <> 0 THEN
592 61              [UF] := [GSUBO(BC)] * [UA];
593 61              IF NRSET <> 0 [AF] := [GSUBO(BC)] * [AA];
594 61          ENDIF;
595 51          IF BMODES <> 0 [PHIF] := [GSUBO(BC)] * [PHIA];
596 51          IF BDTR <> 0 OR BMTR <> 0 [UTRANF] := [GSUBO(BC)] * [UTRANA];
597 51          IF BDFR <> 0 OR BMFR <> 0 [UFREQF] := [GSUBO(BC)] * [UFREQA];
598 51      ELSE

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT  LEVL
599  51
600  61$
601  61$
602  61$
603  61
604  71
605  71
606  71
607  81
608  71
609  61
610  71
611  71
612  71
613  71
614  81
615  71
616  61
617  71
618  71
619  71
620  61
621  71
622  71
623  71
624  61
625  71
626  71
627  71
628  61
629  61$
630  61$
631  61$
632  61
633  71
634  71
635  71
636  61
637  71
638  71
639  71
640  61
641  61
642  61
643  61
644  51
645  41$
646  41
647  41
648  51$
649  51$
650  51$
651  51
652  61
653  61
654  61
655  51
656  61
657  61
658  61
659  51
660  61

IF NOMIT <> 0 THEN
DATA RECOVERY WITH STATIC CONDENSATION
IF BLOAD <> 0 THEN
CALL RECOVA ( [UA[, [PO[, [GSUBO(BC)[, NRSET, [AA[,
               [IFM(BC)[, , [KOOINV(BC)[, [PFOA(BC)[, [UF[ ]
IF NRSET <> 0 CALL RECOVA ( [AA[, , [GSUBO(BC)[, [PFOA(BC)[, [AF[ ]
ENDIF;
IF BSAERO <> 0 THEN
CALL RECOVA ( [UA[, [PO[, [GASUBO(BC)[, NRSET, [AA[,
               [IFM(BC)[, BSAERO, [KOOL(BC)[, [KOOU(BC)[,
               [PFOA(BC)[, [UF[ ]
IF NRSET <> 0 CALL RECOVA ( [AA[, [GASUBO(BC)[, [PFOA(BC)[, [AF[ ]
ENDIF;
IF BMODES <> 0 THEN
[PHIO] := [GSUBO(BC)] * [PHIA[;
CALL ROWMERGE ( [PHIF[, [PHIO[, [PHIA[, [PFOA(BC)[ ]
ENDIF;
IF BDTR <> 0 OR BMTR <> 0 THEN
CALL RECOVA ( [UTRANA[, , [GSUBO(BC)[, [PFOA(BC)[, [UTRANF[ ]
ENDIF;
IF BDFR <> 0 OR BMFR <> 0 THEN
CALL RECOVA ( [UFREQA[, , [GSUBO(BC)[, [PFOA(BC)[, [UFREQF[ ]
ENDIF;
ELSE
DATA RECOVERY WITHOUT F-SET REDUCTION
IF BLOAD <> 0 THEN
[UF] := [UA[;
IF NRSET <> 0 [AF[ := [AA[;
ENDIF;
IF BSAERO <> 0 THEN
[UF] := [UA[;
IF NRSET <> 0 [AF[ := [AA[;
ENDIF;
IF BMODES <> 0 [PHIF[ := [PHIA[;
IF BDTR <> 0 OR BMTR <> 0 [UTRANF[ := [UTRANA[;
IF BDFR <> 0 OR BMFR <> 0 [UFREQF[ := [UFREQA[;
ENDIF;
ENDIF;
IF NUMOPTBC > 1 CALL NULLMAT ( [UN[, [AN[, [PHIN[ ]
IF NSPC <> 0 THEN
DATA RECOVERY WITH SPC-REDUCTION
IF BLOAD <> 0 THEN
CALL YSMERGE ( [UN[, [YS(BC)[, [UF[, [PNSF(BC)[ ]
IF NRSET <> 0 CALL YSMERGE ( [AN[, , [AF[, [PNSF(BC)[ ]
ENDIF;
IF BSAERO <> 0 THEN
CALL YSMERGE ( [UN[, [YS(BC)[, [UF[, [PNSF(BC)[ ]
IF NRSET <> 0 CALL YSMERGE ( [AN[, , [AF[, [PNSF(BC)[ ]
ENDIF;
IF BMODES <> 0 CALL YSMERGE ( [PHIN[, [YS(BC)[, [PHIF[,
                           [PNSF(BC)[ ]

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
661 5!      IF BDTR <> 0 OR BMTR <> 0
662 6!          CALL YSMERGE ( [UTRANN], [YS(BC)], [UTRANF],
663 6!              [PNSF(BC)], BDTR );
664 5!      IF BDFR <> 0 OR BMFR <> 0
665 6!          CALL YSMERGE ( [UFREQN], [YS(BC)], [UFREQF],
666 6!              [PNSF(BC)], BDFR );
667 5!      ELSE
668 5!$
669 5!$      DATA RECOVERY WITHOUT SPC-REDUCTION
670 5!$
671 5!      IF BLOAD <> 0 THEN
672 6!          [UN] := [UF];
673 6!          IF NRSET <> 0 [AN] := [AF];
674 6!      ENDIF;
675 5!      IF BSAERO <> 0 THEN
676 6!          [UN] := [UF];
677 6!          IF NRSET <> 0 [AN] := [AF];
678 6!      ENDIF;
679 5!      IF BMODES <> 0 [PHIN] := [PHIF];
680 5!      IF BDTR <> 0 OR BMTR <> 0 [UTRANN] := [UTRANA];
681 5!      IF BDFR <> 0 OR BMFR <> 0 [UFREQN] := [UFREQA];
682 5!      ENDIF;
683 4!$
684 4!      IF NUMOPTBC > 1 CALL NULLMAT ( [UG(BC)], [AG(BC)], [PHIG(BC)] );
685 4!      IF NMPC <> 0 THEN
686 5!$
687 5!$      DATA RECOVERY WITH MPC-REDUCTION
688 5!$
689 5!      IF BLOAD <> 0 THEN
690 6!          [UM] := [TMN(BC)] * [UN];
691 6!          CALL ROWMERGE ( [UG(BC)], [UM], [UN], [PGMN(BC)] );
692 6!          IF NRSET <> 0 THEN
693 7!              [UM] := [TMN(BC)] * [AN];
694 7!              CALL ROWMERGE ( [AG(BC)], [UM], [AN], [PGMN(BC)] );
695 7!          ENDIF;
696 6!      ENDIF;
697 5!      IF BSAERO <> 0 THEN
698 6!          [UM] := [TMN(BC)] * [UN];
699 6!          CALL ROWMERGE ( [UG(BC)], [UM], [UN], [PGMN(BC)] );
700 6!          IF NRSET <> 0 THEN
701 7!              [UM] := [TMN(BC)] * [AN];
702 7!              CALL ROWMERGE ( [AG(BC)], [UM], [AN], [PGMN(BC)] );
703 7!          ENDIF;
704 6!      ENDIF;
705 5!      IF BMODES <> 0 THEN
706 6!          [UM] := [TMN(BC)] * [PHIN];
707 6!          CALL ROWMERGE ( [PHIG(BC)], [UM], [PHIN], [PGMN(BC)] );
708 6!      ENDIF;
709 5!      IF BDTR <> 0 OR BMTR <> 0 THEN
710 6!          [UM] := [TMN(BC)] * [UTRANN];
711 6!          CALL ROWMERGE ( [UTRANG], [UM], [UTRANN], [PGMN(BC)] );
712 6!      ENDIF;
713 5!      IF BDFR <> 0 OR BMFR <> 0 THEN
714 6!          [UM] := [TMN(BC)] * [UFREQN];
715 6!          CALL ROWMERGE ( [UFREQG], [UM], [UFREQN], [PGMN(BC)] );
716 6!      ENDIF;
717 5!      ELSE
718 5!$

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
719 51$ DATA RECOVERY WITHOUT MPC-REDUCTION $!
720 51$ $!
721 51 IF BLOAD <> 0 THEN !
722 61 [UG(BC)] := [UN]; !
723 61 IF NRSET <> 0 [AG(BC)] := [AN]; !
724 61 ENDIF; !
725 51 IF BSAERO <> 0 THEN !
726 61 [UG(BC)] := [UN]; !
727 61 IF NRSET <> 0 [AG(BC)] := [AN]; !
728 61 ENDIF; !
729 51 IF BMODES <> 0 [PHIG(BC)] := [PHIN]; !
730 51 IF BDTR <> 0 OR BMTR <> 0 [UTRANG] := [UTRANN]; !
731 51 IF BDFR <> 0 OR BMFR <> 0 [UFREQG] := [UFREQN]; !
732 51 ENDIF; !
733 41$ $!
734 41$ RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS $!
735 41$ $!
736 41 IF BBLAST <> 0 [UBLASTG] := [PHIG(BC)] * [UBLASTI]; !
737 41$ $!
738 41$ PERFORM CONSTRAINT EVALUATION FOR STATIC DISCIPLINES $!
739 41$ $!
740 41 CALL DC /AL ( BC, [UG(BC)] ); !
741 41 IF BLOAD <> 0 OR BSAERO <> 0 CALL SCEVAL ( BC, [GLBSIG], [UG(BC)], !
742 51 TRMTYP ); !
743 41$ $!
744 41$ HANDLE OUTPUT REQUESTS $!
745 41$ $!
746 41 CALL OPFLOAD ( NUMOPTBC, BC, NITER, GSIZE, [PG], TRMTYP, QDP, !
747 41 [GTKG], [AIRFRM(MINDEX)], [DELTA] ); !
748 41 CALL OPFDISP ( NUMOPTBC, BC, NITER, GSIZE, [UG(BC)], [AG(BC)], !
749 41 TRMTYP, [UG(BC)], [AG(BC)], [UBLASTG], , [UTRANG], !
750 41 [UTRANE], [UFREQG], [UFREQE], LAMBDA, [PHIG(BC)] ); !
751 41 CALL EDR ( NUMOPTBC, BC, NITER, NDV, GSIZE, EOSUMMARY, EODISC, !
752 41 [UG(BC)], [UG(BC)], , [UTRANG], [UFREQG], [PHIG(BC)] ); !
753 41 CALL OPFEDR ( BC, HSIZE, NITER, TRMTYP ); !
754 41 ENDDO; !
755 31$ $!
756 31$ SELECT ACTIVE CONSTRAINTS $!
757 31$ $!
758 31 NRFAC := 3.0; !
759 31 EPS := - 0.10; !
760 31 CALL ACTCON ( NITER, MAXITER, NRFAC, NDV, EPS, CONVERGE, !
761 31 CTL, CTLMIN, [AMAT] ); !
762 31$ $!
763 31 IF CONVERGE < 2 AND NITER <= MAXITER THEN !
764 41$ $!
765 41$ CHECK FOR FSD ITERATIONS $!
766 41$ $!
767 41 CALL FSD ( NDV, NITER, MAXFSD, OPSTRAT, ALPHA, FSDFLG, !
768 41 CNVRGLIM, CONVERGE, CTL, CTLMIN ); !
769 41$ $!
770 41 IF FSDFLG = 0 THEN !
771 51$ $!
772 51$ USE MATHEMATICAL PROGRAMMING METHODS $!
773 51$ OBTAIN THE SENSITIVITIES OF THE CONSTRAINTS WRT THE $!
774 51$ DESIGN VARIABLES $!
775 51$ $!
776 51 CALL MAKDFV (NDV, [AMAT], MOVLIM ); !
777 51$ $!

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
778 5!$*****:~$!
779 5!$ SENSITIVITY EVALUATION FOR BOUNDARY CONDITION DEPENDENT CONSTRAINTS$:~$!
780 5!$*****:~$!
781 5!$~$!
782 5!$ FOR BC = 1 TO NUMOPTBC DO~$!
783 6!$ CALL ABOUND ( BC, ABC, PCA, NAU, NACSD, [PGA], ADC, AFC, AAC,~$!
784 6!$ TRMTYP, MINDEX, NAE, NAUE, PAE,~$!
785 6!$ NMPC, NSPC, NOMIT, NRSET, NGDR, ['G(BC)] );~$!
786 6!$ IF ABC > 0 THEN~$!
787 7!$~$!
788 7!$ EVALUATE FREQUENCY CONSTRAINT SENSITIVITIES~$!
789 7!$~$!
790 7!$ IF ADC <> 0 CALL FREQSNS ( BC, ADC, NDV, [PHIG(BC)], [AMAT] );~$!
791 7!$~$!
792 7!$ EVALUATE FLUTTER CONSTRAINT SENSITIVITIES~$!
793 7!$~$!
794 7!$ IF AFC <> 0 CALL FLUTSENS ( BC, GSIZE, NDV, [QHHL(BC)],~$!
795 8!$ [MTH(BC)], [KHHF(BC)],~$!
796 8!$ [PHIG(BC)], [AMAT] );~$!
797 7!$ IF NAU > 0 THEN~$!
798 8!$~$!
799 8!$ SENSITIVITIES OF CONSTRAINTS WRT DISPLACEMENTS~$!
800 8!$~$!
801 8!$ CALL NULLMAT ( [DFDU], [DPGV] );~$!
802 8!$ IF NACSD > NAU * NDV OR AAC <> 0 THEN~$!
803 9!$~$!
804 9!$ USE GRADIENT METHOD~$!
805 9!$~$!
806 9!$ CALL MAKDFU ( BC, GSIZE, [DFDU] );~$!
807 9!$ ELSE~$!
808 9!$~$!
809 9!$ USE VIRTUAL LOAD METHOD~$!
810 9!$~$!
811 9!$ CALL MAKDFU ( BC, GSIZE, [DPGV] );~$!
812 9!$ ENDIF;~$!
813 8!$~$!
814 8!$ SOME RELATIVELY SIMPLE CALCULATIONS THAT PRECEDE THE~$!
815 8!$ LOOP ON THE DESIGN VARIABLES~$!
816 8!$~$!
817 8!$ CALL COLPART ( [UG(BC)], , [UGA], [PGA] );~$!
818 8!$~$!
819 8!$ OBTAIN THE SENSITIVITIES OF THE DESIGN~$!
820 8!$ DEPENDENT LOADS~$!
821 8!$~$!
822 8!$ CALL DDLOAD(NDV, GSIZE, BC, DDPLG, [PGA], [DPVJ]);~$!
823 8!$~$!
824 8!$ CALL MAKDVU ( NDV, [UGA], [DKUG], GMMCT, DMVI );~$!
825 8!$ CALL NULLMAT ( [DUG] );~$!
826 8!$ IF NRSET <> 0 THEN~$!
827 9!$ CALL COLPART ( [AG(BC)], , [AGA], [PGA] );~$!
828 9!$ CALL MAKDVU ( NDV, [AGA], [DMAG], GMMCT, DMVI );~$!
829 9!$ [DUG] := [DKUG] + [DMAG];~$!
830 9!$ CALL MAKDVU ( NDV, [UGA], [DMUG], GMMCT, DMVI );~$!
831 9!$ ELSE~$!
832 9!$ [DUG] := [DKUG];~$!
833 9!$ ENDIF;~$!
834 8!$~$!
835 8!$ ACCOUNT FOR VIRTUAL LOAD METHOD~$!
836 8!$~$!

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVEL
837 81      IF NACSD > NAU * NDV OR AAC <> 0 THEN      !
838 91$                                           $!
839 91$      USE GRADIENT METHOD                      $!
840 91$                                           $!
841 91      IF DDPLG > 0 THEN                        !
842 101      [DPGV] := [DPVJ] + [DUG];              !
843 101      ELSE                                    !
844 101      [DPGV] := [DUG];                        !
845 101      ENDIF;                                  !
846 91      ELSE                                    !
847 91$                                           $!
848 91$      USE VIRTUAL LOAD METHOD                  $!
849 91$                                           $!
850 91      IF DDPLG > 0 THEN                        !
851 101      [DPDU] := [DPVJ] + [DUG];              !
852 101      ELSE                                    !
853 101      [DPDU] := [DUG];                        !
854 101      ENDIF;                                  !
855 91      ENDIF;                                  !
856 81$                                           $!
857 81$      REDUCE THE RIGHT HAND SIDES TO THE L SET $!
858 81$                                           $!
859 81      CALL NULLMAT ( [DPNV], [DMUN] );          !
860 81      IF NMPC <> 0 THEN                        !
861 91      CALL GREduce ( , [DPGV], [PGMN(BC)], [TMN(BC)], , [DPNV] ); !
862 91      IF NRSET <> 0                            !
863 101      CALL GREduce ( , [DMUG], [PGMN(BC)], [TMN(BC)], , [DMUN] ); !
864 91      ELSE                                    !
865 91      [DPNV] := [DPGV];                        !
866 91      IF NRSET <> 0 [DMUN] := [DMUG];          !
867 91      ENDIF;                                  !
868 81$                                           $!
869 81      CALL NULLMAT ( [DPFV], [DMUF] );          !
870 81      IF NSPC <> 0 THEN                        !
871 91      CALL NREDUCE ( , [DPNV], [PNSF(BC)], , , , [DPFV] );      !
872 91      IF NRSET <> 0                            !
873 101      CALL NREDUCE ( , [DMUN], [PNSF(BC)], , , , [DMUF] );      !
874 91      ELSE                                    !
875 91      [DPFV] := [DPGV];                        !
876 91      IF NRSET <> 0 [DMUF] := [DMUN];          !
877 91      ENDIF;                                  !
878 81$                                           $!
879 81      CALL NULLMAT ( [DPAV], [DMUA] );          !
880 81      IF NGDR <> 0 THEN                        !
881 91      [DPAV] := TRANS( [GSUBO(BC)] ) * [DPFV]; !
882 91      IF NRSET <> 0 [DMUA] := TRANS( [GSUBO(BC)] ) * [DMUF]; !
883 91      ELSE                                    !
884 91      IF NOMIT <> 0 THEN                        !
885 101      IF AAC <> 0 THEN                        !
886 111      CALL FREDUCE ( , [DPFV], [PFOA(BC)], AAC, !
887 111      [KOOL(BC)], [KOOU(BC)],                !
888 111      [KAO(BC)], [GASUBO(BC)], ,              !
889 111      [DPAV], [DPOV] );                        !
890 111      IF NRSET <> 0                            !
891 121      CALL FREDUCE ( , [DMUF], [PFOA(BC)], AAC, !
892 121      [KOOL(BC)], [KOOU(BC)],                !
893 121      [KAO(BC)], [GASUBO(BC)], ,              !
894 121      [DMUA], [DMUO] );                        !
895 111      ELSE                                    !
896 111      CALL FREDUCE ( , [DPFV], [PFOA(BC)], ,    !
897 111      [KOOINV(BC)], , , [GSUBO(BC)], ,        !
898 111      [DPAV], [DPOV] );                        !

```

Figure C.1 Standard MAPOL Sequence (Continued)



\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVEL
899 11: IF NRSET <> 0
900 12: CALL FREDUCE ( , [DMUF], [PFOA(BC)], ,
901 12: [KOOINV(BC)],,, [GSUBO(BC)], ,
902 12: [DMUA], [DMUO] );
903 11: ENDIF;
904 10: ELSE
905 10: [DPAV] := [DPFV];
906 10: IF NRSET <> 0 [DMUA] := [DMUF];
907 10: ENDIF;
908 9: ENDIF;
909 8: $!
910 8: IF NRSET <> 0 THEN
911 9: CALL ROWPART ( [DPAV], [DPRV], [DPLV], [PARL(BC)] );
912 9: CALL ROWPART ( [DMUA], [DMUR], [DMUL], [PARL(BC)] );
913 9: [DMU] := TRANS([D(BC)]) * [DMUL] + [DMUR];
914 9: CALL ROWMERGE ( [DP1], [DMU], [DPLV], [PARL(BC)] );
915 9: [DRHS] := TRANS( [D(BC)] ) * [DPLV] + [DPRV];
916 9: IF AAC <> 0 THEN
917 10: $!
918 10: PROCESS ACTIVE CONSTRAINTS FOR SAERO DISCIPLINE
919 10: $!
920 10: CALL GFBS ( [KL11(BC)], [KU11(BC)], [DP1], [DK1V] );
921 10: [DRHS] := [DRHS] - [K21(BC)] * [DK1V];
922 10: $!
923 10: CALL AEROSENS TO
924 10: $!
925 10: PROCESS ACTIVE AERO EFFECTIVENESS CONSTRAINTS.
926 10: $!
927 10: NOTE THAT NAU IS DECREMENTED IN AEROSENS TO
928 10: $!
929 10: ACCOUNT FOR PSEUDO DISPLACEMENTS ASSOCIATED WITH
930 10: $!
931 10: AEROELASTIC CONTROL EFFECTIVENESS CONSTRAINTS
932 10: $!
933 10: CALL AEROSENS (BC, TRMTYP, MINDEX, NDV, NAU, NAE,
934 10: $!
935 10: NAUE, PAE, [DK1V], [DRHS], [KL112(BC)],
936 10: $!
937 10: [K21(BC)], [RHS(BC)], [LHS(BC)], [PAR(BC)],
938 10: $!
939 10: [DUAV], [PGA], [DDELTV], [AMAT] );
940 10: ELSE
941 10: $!
942 10: PROCESS ACTIVE CONSTRAINTS FOR STATICS DISCIPLINE
943 10: $!
944 10: CALL INERTIA ( [MRR(BC)], [DRHS], [DURD], 1 );
945 10: $!
946 10: [DULD] := [D(BC)] * [DURD];
947 10: $!
948 10: CALL ROWMERGE ( [DUAD], [DURD], [DULD], [PARL(BC)] );
949 10: $!
950 10: [DPLV] := [DPLV] + [IFR(BC)] * [DURD];
951 10: $!
952 10: CALL FBS ( [KLLINV(BC)], [DPLV], [DULV] );
953 10: $!
954 10: CALL YSMERGE ( [DUAV], , [DULV], [PARL(BC)] );
955 10: $!
956 10: ENDIF;
957 10: $!
958 9: ELSE
959 9: $!
960 9: NOTE THAT SAERO W/O SUPPORT IS NOT SUPPORTED
961 9: $!
962 9: CALL FBS ( [KLLINV(BC)], [DPAV], [DUAV] );
963 9: $!
964 9: ENDIF;
965 8: $!
966 8: CONTINUE WITH THE RECOVERY ONLY IF THERE ARE STILL ACTIVE
967 8: $!
968 8: DISPLACEMENTS. IF THE ONLY DISPLACEMENT DEPENDENT
969 8: $!
970 8: CONSTRAINTS ARE AEROELASTIC CONTROL EFFECTIVENESS
971 8: $!
972 8: CONSTRAINTS, THEN THERE IS NO NEED TO CONTINUE WITH THE
973 8: $!
974 8: RECOVERY PROCESS. PROCEED WITH THE NEXT ACTIVE BOUNDARY
975 8: $!
976 8: CONDITION.
977 8: $!

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

STAT	LEVEL		
958	81	IF NAU <> 0 THEN	!
959	91\$		\$!
960	91\$	RECOVER TO THE F SET	\$!
961	91\$		\$!
962	91	CALL NULLMAT ( [DUFV] );	!
963	91	IF NGDR <> 0 THEN	!
964	101	[DUFV] := [GSUBO(BC)] * [DUAV];	!
965	101	ELSE	!
966	101	IF NOMIT <> 0 THEN	!
967	111	IF NRSET <> 0 THEN	!
968	121	IF AAC <> 0 THEN	!
969	131	[TMP1] := [DPOV] + [POARO(BC)] * [DDELDV];	!
970	131	ELSE	!
971	131	[TMP1] := [DPOV] - [IFM(BC)] * [DUAD];	!
972	131	ENDIF;	!
973	121	ELSE	!
974	121	[TMP1] := [DPOV];	!
975	121	ENDIF;	!
976	111	IF AAC <> 0 THEN	!
977	121	CALL GFBS ( [KOOL(BC)], [KOOU(BC)], [TMP1],	!
978	121	[UOO] );	!
979	121	[UO] := [GASUBO(BC)] * [DUAV] + [UOO];	!
980	121	ELSE	!
981	121	CALL FBS ( [KOOINV(BC)], [TMP1], [UOO] );	!
982	121	[UO] := [GSUBO(BC)] * [DUAV] + [UOO];	!
983	121	ENDIF;	!
984	111	CALL ROWMERGE ([DUFV], [UO], [DUAV], [PFOA(BC)]);	!
985	111	ELSE	!
986	111	[DUFV] := [DUAV];	!
987	111	ENDIF;	!
988	101	ENDIF;	!
989	91\$		\$!
990	91\$	REDUCE THE LEFT HAND SIDE MATRIX	\$!
991	91\$		\$!
992	91	IF NMPC <> 0 THEN	!
993	101	CALL GREduce (,[DFDU],[PGMN(BC)],[TMN(BC)],,[DFDUN]);	!
994	101	ELSE	!
995	101	[DFDUN] := [DFDU];	!
996	101	ENDIF;	!
997	91\$		\$!
998	91	IF NSPC <> 0 THEN	!
999	101	CALL ROWPART ( [DFDUN], , [DFDUF], [PNSF(BC)] );	!
1000	101	ELSE	!
1001	101	[DFDUF] := [DFDUN];	!
1002	101	ENDIF;	!
1003	91\$		\$!
1004	91\$	ACCOUNT FOR VIRTUAL LOAD METHOD	\$!
1005	91\$		\$!
1006	91	IF NACSD > NAU * NDV OR AAC <> 0 THEN	!
1007	101\$		\$!
1008	101\$	USE GRADIENT METHOD	\$!
1009	101\$		\$!
1010	101	CALL MKAMAT ([AMAT], [DFDUF], [DUFV], PCA, [PGA] );	!
1011	101	ELSE	!
1012	101\$		\$!
1013	101\$	USE VIRTUAL LOAD METHOD	\$!
1014	101\$		\$!
1015	101	CALL MKAMAT ([AMAT], [DUFV], [DFDUF], PCA, [PGA] );	!
1016	101	ENDIF;	!
1017	91\$		\$!
1018	91	ENDIF; \$ END IF ON REMAINING ACTIVE APPLIED LOADS	\$!
1019	81	ENDIF; \$ END IF ON ACTIVE APPLIED LOADS	\$!
1020	71	ENDIF; \$ END IF ON ACTIVE BOUNDARY CONDITION	\$!
1021	61	ENDDO; \$ END DO ON ACTIVE BOUNDARY CONDITIONS	\$!

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
1022 51$
1023 51$      DEFINE A MOVE LIMIT IN DESIGN:
1024 51$      VMIN = V / MOVLM OR MINIMUM D.V. VALUE
1025 51$      VMAX = V * MOVLM OR MAXIMUM D.V. VALUE
1026 51$      MOVLM GREATER THAN 1.0
1027 51$
1028 51      CALL DESIGN( CONVERGE, MOVLM, CNVRGLIM, CTL, CTLMIN, OPSTRAT,
1029 51      NUMOPTBC, [AMAT] );
1030 51$
1031 51      ENDIF; $      END IF ON FSDPLG TEST
1032 41      ENDIF; $      END IF TEST AFTER ACTCON
1033 31      ENDDO; $      END WHILE LOOP FOR CONVERGE
1034 21ENDIF; $      END IF ON OPTIMIZATION
1035 11$
1036 11$*****
1037 11$      BEGIN FINAL ANALYSIS LOOP
1038 11$*****
1039 11$
1040 11IF NBNDCOND > NUMOPTBC THEN
1041 21$
1042 21$      ASSEMBLE THE GLOBAL MATRICES
1043 21$
1044 21      NEG1 := -1;
1045 21      CALL EMA2 ( GSIZE, [KGG], [MGG] );
1046 21      FOR BC = NUMOPTBC + 1 TO NBNDCOND DO
1047 31      CALL BOUND [BC, NMPC, NSPC, NOMIT, NRSET, NGDR ];
1048 31      CALL BDCASE (BC, BLOAD, BMAS, BMODES, BSAERO, QDP, MINDEX,
1049 31      SYM, TRMTYP, BFLUTR, BDRSP, BDTR, BMTR, BDFR, BMFR,
1050 31      BGUST, BBLAST );
1051 31      IF BLOAD <> 0 CALL GTLOAD ( BC, GSIZE, [PG] );
1052 31      IF BSAERO <> 0 THEN
1053 41      CALL NULLMAT ( [AIC] );
1054 41      IF SYM = 1 THEN
1055 51      [AIC] := [AICMAT(MINDEX)];
1056 51      ELSE IF SYM = -1 THEN
1057 61      [AIC] := [AAICMAT(MINDEX)];
1058 61      ENDIF;
1059 51      ENDIF;
1060 41      ENDIF;
1061 31$
1062 31$      PARTITION-REDUCTION OF GLOBAL MATRICES
1063 31$
1064 31      IF NBNDCOND > 1 CALL NULLMAT ( [KNN], [PN], [MNN], [GTKN], [GSTKN],
1065 41      [UGTKN] );
1066 31      IF NMPC <> 0 THEN
1067 41$
1068 41$      PERFORM MPC REDUCTION
1069 41$
1070 41      CALL GREduce ( [KGG], [PG], [PGMN(BC)], [TMN(BC)], [KNN], [PN] );
1071 41      IF BMAS <> 0 CALL GREduce ([MGG], [PGMN(BC)], [TMN(BC)], [MNN]);
1072 41      IF BSAERO <> 0 THEN
1073 51      CALL GREduce (, [GTKG], [PGMN(BC)], [TMN(BC)],, [GTKN]);
1074 51      CALL GREduce (, [GSTKG], [PGMN(BC)], [TMN(BC)],, [GSTKN]);
1075 51      ENDIF;
1076 41      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0
1077 51      CALL GREduce (, [UGTKG], [PGMN(BC)], [TMN(BC)],, [UGTKN] );
1078 41      ELSE

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
1079 41$
1080 41$ NO MPC REDUCTION $1
1081 41$ $1
1082 41 [KNN] := [KGG]; 1
1083 41 IF BLOAD <> 0 [PN] := [PG]; 1
1084 41 IF BMASS <> 0 [MNN] := [MGG]; 1
1085 41 IF BSAERO <> 0 THEN 1
1086 51 [GTKN] := [GTKG]; 1
1087 51 [GSTKN] := [GSTKG]; 1
1088 51 ENDIF; 1
1089 41 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKN] := [UGTKG]; 1
1090 41 ENDIF; 1
1091 31 IF NBNDCOND > 1 CALL NULLMAT ( [KFF], [PF], [MFF], [GTKF], [GSTKF], 1
1092 41 [UGTKF] ); 1
1093 31 IF NSPC <> 0 THEN 1
1094 41$ $1
1095 41$ PERFORM SPC REDUCTION $1
1096 41$ $1
1097 41 CALL NREDUCE ( [KNN], [PN], [PNSF(BC)], [YS(BC)], [KFF], [KFS], 1
1098 41 [KSS], [PF] ); 1
1099 41 IF BMASS <> 0 CALL NREDUCE ( [MNN], , [PNSF(BC)], , [MFF]); 1
1100 41 IF BSAERO <> 0 THEN 1
1101 51 CALL NREDUCE ( , [GTKN], [PNSF(BC)], , , , [GTKF] ); 1
1102 51 CALL NREDUCE ( , [GSTKN], [PNSF(BC)], , , , [GSTKF] ); 1
1103 51 ENDIF; 1
1104 41 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 1
1105 51 CALL NREDUCE ( , [UGTKN], [PNSF(BC)], , , , [UGTKF]); 1
1106 41 ELSE 1
1107 41$ $1
1108 41$ NO SPC REDUCTION $1
1109 41$ $1
1110 41 [KFF] := [KNN]; 1
1111 41 IF BLOAD <> 0 [PF] := [PN]; 1
1112 41 IF BMASS <> 0 [MFF] := [MNN]; 1
1113 41 IF BSAERO <> 0 THEN 1
1114 51 [GTKF] := [GTKN]; 1
1115 51 [GSTKF] := [GSTKN]; 1
1116 51 ENDIF; 1
1117 41 IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKF] := [UGTKN]; 1
1118 41 ENDIF; 1
1119 31$ $1
1120 31$ IF STEADY AERODYNAMICS FOR THIS B.C., ADJUST THE KFF MATRIX $1
1121 31$ $1
1122 31 IF BSAERO <> 0 THEN 1
1123 41 CALL TRNPOSE ( [GSTKF], [GSKF] ); 1
1124 41 [AICS] := [GTKF] * [ TRANS([AIC]) * [GSKF] ]; 1
1125 41 [PAF] := (QDP) [ [GTKF] * [AIRFRM(MINDEX)] ]; 1
1126 41 [KAPF] := [KFF] - (QDP) [AICS]; 1
1127 41 ENDIF; 1
1128 31$ $1
1129 31 IF NBNDCOND > 1 CALL NULLMAT ([KAA], [PA], [MAA], [KAAA], [PAA], [UGTKA]); 1
1130 31 IF NGDR <> 0 THEN 1
1131 41$ $1
1132 41$ PERFORM THE GENERAL DYNAMIC REDUCTION $1
1133 41$ $1
1134 41 IF NOMIT <> 0 OR NRSET <> 0 THEN 1
1135 51$ $1
1136 51$ OBTAIN THE OMITTED DOF PARTITION OF KFF AND MFF $1
1137 51$ $1
1138 51 CALL PARTN ( [KFF], [KOO], , [KOA], , [PFOA(BC)] ); 1
1139 51 CALL PARTN ( [MFF], [MOO], , , [PFOA(BC)] ); 1

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
1140 51      ELSE
1141 51      [KOO] := [KFF];
1142 51      [MOO] := [MFF];
1143 51      ENDIF;
1144 41      CALL GDR1 ([KOO], [MOO], [KSOO], [GGO], LKSET, LJSET, NEIV, FMAX, BC);
1145 41$
1146 41$      LKSET      MEANING
1147 41$      <> 0      K-SET EXISTS
1148 41$      = 0      K-SET DOES NOT EXIST
1149 41$
1150 41      IF LKSET <> 0 THEN
1151 51      CALL SDCOMP ( [KSOO], [LSOO] );
1152 51      CALL GDR2 ( [LSOO], [MOO], [PHIOK], LKSET, LJSET, NEIV, FMAX, BC);
1153 51      ENDIF;
1154 41      CALL GDR3 ( [KOO], [KOA], [MGG], [PHIOK], [TMN(BC)], [GGO],
1155 41      [PGMN(BC)], [PNSF(BC)], [PFOA(BC)],
1156 41      [GSUBO(BC)], LKSET, LJSET, NOMIT, NRSET, GNORM, BC );
1157 41      CALL GDR4 ( BC, GSIZE, GSIZE, LKSET, LJSET, NUMOPTBC,
1158 41      NBNDCOND, [GDRGO(BC)], , , , , [PARL(BC)] );
1159 41$
1160 41      [MAA] := TRANS ( [GSUBO(BC)] ) * [ [MFF] * [GSUBO(BC)] ];
1161 41      [KAA] := TRANS ( [GSUBO(BC)] ) * [ [KFF] * [GSUBO(BC)] ];
1162 41      IF BLOAD <> 0 [PA] := TRANS ( [GSUBO(BC)] ) * [PF];
1163 41      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
1164 51      [TMP1] := TRANS ( [UGTKF] ) * [GSUBO(BC)];
1165 51      CALL TRNSPOSE ( [TMP1], [UGTKA] );
1166 51      ENDIF;
1167 41      ELSE
1168 41      IF NOMIT <> 0 THEN
1169 51$      PERFORM THE STATIC REDUCTION
1170 51$
1171 51$      IF BSAERO <> 0 THEN
1172 51      IF BLOAD = 0 THEN
1173 61      FORM [KAA] SO [D] CAN BE FORMED
1174 71$
1175 71$      CALL FREDUCE ([KFF], , [PFOA(BC)], , [KOOINV(BC)], , ,
1176 71$      [GSUBO(BC)], [KAA] );
1177 71      ENDIF;
1178 71      CALL FREDUCE ( [KAF], [PAF], [PFOA(BC)], BSAERO, [KOOL(BC)],
1179 61      [KOOV(BC)], [KAO(BC)], [GASUBO(BC)], [KAAA],
1180 61      [PAA], [POARO(BC)] );
1181 61$
1182 61$      PERFORM GUYAN REDUCTION OF THE MASS MATRIX
1183 61$
1184 61$      IF BMASS <> 0 THEN
1185 71      CALL PARTN ( [MFF], [MOO], , [MOA], [MAABAR], [PFOA(BC)] );
1186 71      [MAA] := [MAABAR] + TRANS([MOA]) * [GASUBO(BC)] +
1187 71      TRANS([GASUBO(BC)]) * [MOA] +
1188 71      TRANS([GASUBO(BC)]) * [ [MOO] * [GASUBO(BC)] ];
1189 71      IF NRSET <> 0 [IFM(BC)] := [MOO] * [GASUBO(BC)] + [MOA];
1190 71      ENDIF;
1191 61      ELSE
1192 61      CALL FREDUCE ( [KFF], [PF], [PFOA(BC)], , [KOOINV(BC)], , ,
1193 61      [GSUBO(BC)], [KAA], [PA], [PO] );
1194 61$
1195 61$      PERFORM GUYAN REDUCTION OF THE MASS MATRIX
1196 61$
1197 61$
1198 61$

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
1199 6!      IF BMAS <> 0 THEN
1200 7!          CALL PARTN ( [MFF], [MOO], , [MOA], [MAABAR], [PFOA(BC)] );
1201 7!          [MAA] := [MAABAR] + TRANS([MOA]) * [GSUBO(BC)] +
1202 7!              TRANS([GSUBO(BC)]) * [MOA] +
1203 7!              TRANS([GSUBO(BC)]) * [MOO] * [GSUBO(BC)] ;
1204 7!          IF NRSET <> 0 [IFM(BC)] := [MOO] * [GSUBO(BC)] + [MOA];
1205 7!      ENDIF;
1206 6!      ENDIF;
1207 5!      IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 THEN
1208 6!          CALL ROWPART ([UGTKF], [UGTKO], [UGTKAB], [PFOA(BC)]);
1209 6!          [TMP1] := TRANS([UGTKO]) * [GSUBO(BC)];
1210 6!          CALL TRNSPOSE ( [TMP1], [TMP2] );
1211 6!          [UGTKA] := [UGTKAB] + [TMP2];
1212 6!      ENDIF;
1213 5!      ELSE
1214 5!$
1215 5!$      NO F-SET REDUCTION
1216 5!$
1217 5!          [KAA] := [KFF];
1218 5!          IF BLOAD <> 0 [PA] := [PF];
1219 5!          IF BSAERO <> 0 THEN
1220 6!              [KAAA] := [KAFF];
1221 6!              [PAA] := [PAF];
1222 6!          ENDIF;
1223 5!          IF BFLUTR <> 0 OR BGUST <> 0 OR BBLAST <> 0 [UGTKA] := [UGTKF];
1224 5!          IF BMAS <> 0 [MAA] := [MFF];
1225 5!      ENDIF;
1226 4!      ENDIF;
1227 3!      IF NRSET <> 0 THEN
1228 4!$
1229 4!$      PERFORM THE SUPPORT SET REDUCTION
1230 4!$
1231 4!          CALL PARTN ( [KAA], , [KLR], , [KLL], [PARL(BC)] );
1232 4!          CALL SDCOMP ( [KLL], [KLLINV(BC)] );
1233 4!          CALL FBS ( [KLLINV(BC)], [KLR], [D(BC)], NEG1 );
1234 4!$
1235 4!$      CALCULATE THE REDUCED MASS MATRIX
1236 4!$
1237 4!          CALL PARTN ([MAA], [MRRBAR], [MLR], , [MLL], [PARL(BC)]);
1238 4!          [IFR(BC)] := [MLL] * [D(BC)] + [MLR];
1239 4!          [MRR(BC)] := [MRRBAR] + TRANS ( [MLR] ) * [D(BC)] +
1240 4!              TRANS ( [D(BC)] ) * [IFR(BC)];
1241 4!          [R22] := TRANS ( [D(BC)] ) * [MLR] + [MRRBAR];
1242 4!          CALL TRNSPOSE ( [IFR(BC)], [R21] );
1243 4!          IF BSAERO <> 0 THEN
1244 5!$
1245 5!$      PROCESS STEADY AEROELASTIC DISCIPLINE
1246 5!$
1247 5!          CALL PARTN ( [KAAA], [KARR], [KALR], [KARL], [KALL], [PARL(BC)] );
1248 5!          [R32] := TRANS([D(BC)]) * [KALR] + [KARR];
1249 5!          [R31] := TRANS([D(BC)]) * [KALL] + [KARL];
1250 5!          CALL MERGE ( [K11], [R22], [KALR], [R21], [KALL], [PARL(BC)] );
1251 5!          CALL ROWPART ( [PAA], [PARBAR], [PAL], [PARL(BC)] );
1252 5!          CALL YSMERGE ( [P1], , [PAL], [PARL(BC)] );
1253 5!          CALL YSMERGE ( [K12(BC)], , [IFR(BC)], [PARL(BC)] );
1254 5!          CALL COLMERGE ( [K21(BC)], [R32], [R31], [PARL(BC)] );
1255 5!          CALL DECOMP ( [K11], [K11(BC)], [KU11(BC)] );
1256 5!          CALL GFBS ([K11(BC)], [KU11(BC)], [K12(BC)], [K1112(BC)], NEG1);
1257 5!          CALL GFBS ([K11(BC)], [KU11(BC)], [P1], [PAR(BC)] );
1258 5!          [LHS(BC)] := [MRR(BC)] + [K21(BC)] * [K1112(BC)];
1259 5!          [P2] := [PARBAR] + TRANS([D(BC)]) * [PAL];
1260 5!          [RHS(BC)] := [P2] - [K21(BC)] * [PAR(BC)];

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
1261 51      CALL SAERO ( BC, [LHS(BC)], [RHS(BC)], [AR], [DELTA],
1262 51      [P2], [MRR(BC)] );
1263 51
1264 61      IF TRMTYP > 0 THEN
1265 61          [AL] := [D(BC)] * [AR];
1266 61      CALL ROWMERGE ( [AA], [AR], [AL], [PARL(BC)] );
1267 61      [UA] := [K112(BC)] * [AR] + [PAR(BC)] * [DELTA];
1268 61      IF NOMIT <> 0 [PO] := [POARO(BC)] * [DELTA];
1269 51      ENDIF;
1270 41$
1271 41      IF BLOAD <> 0 THEN
1272 51$
1273 51$      PROCESS STATICS WITH INERTIA RELIEF
1274 51$
1275 51      CALL MERGE ( [K11], [R22], [KLR], [R21], [KLL], [PARL(BC)] );
1276 51      CALL YSMERGE ( [P1], , [PLBAR], [PARL(BC)] );
1277 51      CALL YSMERGE ( [K12(BC)], , [IFR(BC)], [PARL(BC)] );
1278 51      CALL DECOMP ( [K11], [K11(BC)], [KU11(BC)] );
1279 51      [P1] := [P1] - [K12(BC)] * [AR];
1280 51      CALL GFBS ( [K11(BC)], [KU11(BC)], [P1], [UA], NEG1 );
1281 51      [LHS(BC)] := [MRR(BC)];
1282 51      CALL ROWPART ( [PA], [PR], [PLBAR], [PARL(BC)] );
1283 51      [RHS(BC)] := TRANS( [D(BC)] ) * [PLBAR] + [PR];
1284 51      CALL INERTIA ([LHS(BC)], [RHS(BC)], [AR] );
1285 51      [AL] := [D(BC)] * [AR];
1286 51      CALL ROWMERGE ( [AA], [AR], [AL], [PARL(BC)] );
1287 51      [P1] := [P1] - [K12(BC)] * [AR];
1288 51      CALL GFBS ( [K11], [KU11], [P1], [UA] );
1289 51      ENDIF;
1290 41      IF BMODES <> 0 CALL REIG ( BC, [KAA], [MAA], [MRR(BC)], [D(BC)],
1291 51      LAMBDA, [PHIA], [MII], HSIZE);
1292 41      ELSE
1293 41$
1294 41$      NO SUPPORT SET REDUCTION
1295 41$
1296 41      IF BLOAD <> 0 THEN
1297 51      CALL SDCOMP ( [KAA], [KLLINV(BC)] );
1298 51      CALL FBS ( [KLLINV(BC)], [PA], [UA] );
1299 51      ENDIF;
1300 41      IF BSAERO <> 0 THEN
1301 51$
1302 51$      NOT FUNCTIONAL, REQUIRES [DELTA] TO BE INPUT
1303 51$
1304 51      CALL DECOMP ( [KAAA], [KLLL(BC)], [KLLU(BC)] );
1305 51      CALL GFBS ( [KLLL(BC)], [KLLU(BC)], [PAA], [UA] );
1306 51      ENDIF;
1307 41      IF BMODES <> 0 CALL REIG ( BC, [KAA], [MAA],,, LAMBDA, [PHIA],
1308 51      [MII], HSIZE );
1309 41      ENDIF;
1310 31$
1311 31$      PERFORM ANY DYNAMIC ANALYSES -- NOTE THAT THESE ARE INDEPENDENT
1312 31$      OF THE SUPPORT SET
1313 31$
1314 31      IF BDYN <> 0 THEN
1315 41      CALL QHHLGEN (BC, [QKKL], [QKJL], [UGTKA], [PHIA],
1316 41      [PHIKH], [QHHL(BC)], [QHJL]);
1317 41      CALL DMA ( BC, GSIZE, [MAA], [KAA], [TMN(BC)], [GSUBO(BC)], NGDR,
1318 41      LAMBDA, [PHIA], [MDD], [BDD], [KDDT], [KDDF],
1319 41      [MHH(BC)], [BHH], [KHHT], [KHHT(BC)] );

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT LEVL
1320 41 CALL DYNLOAD ( BC, GSIZE, [TMN(BC)], [GSUBO(BC)], NGDR,
1321 41 [PHIA], [QHJL], [PDT], [PDF] );
1322 41 IF BFLUTR <> 0 CALL FLUTTRAN (BC, [QHHL(BC)], LAMBDA, HSIZE,
1323 51 [MH(BC)], [KHHP(BC)]);
1324 41 IF BDRSP <> 0 THEN
1325 51 CALL DYNRSP (BC, [MDD], [BDD], [KDDT], [KDDF], [MH(BC)], [BHH],
1326 51 [KHHT], [KHHP(BC)], [PDT], [PDF], [QHHL(BC)],
1327 51 [UTRANA], [UFREQA], [UTRANI], [UFREQI], [UTRANE],
1328 51 [UFREQE] );
1329 51 IF BMTR <> 0 [UTRANA] := [PHIA] * [UTRANI];
1330 51 IF BMFR <> 0 [UFREQA] := [PHIA] * [UFREQI];
1331 51 ENDIF;
1332 41 ENDIF;
1333 31 IF BBLAST <> 0 THEN
1334 41 CALL BLASTFIT ( BC, [QJJL], [MATTR], [MATSS], [BQDP], [BFRC],
1335 41 [DWNWSH], HSIZE, [ID2], [MPART], [UGTKA],
1336 41 [BLGTJA], [BLSTJA] );
1337 41 CALL COLPART ( [PHIA], , [PHIE], [MPART] );
1338 41 CALL ROMMERGE ( [PHIR], [ID2], [D(BC)], [PARL(BC)] );
1339 41 CALL COLMERGE ( [PHIB], [PHIR], [PHIE], [MPART] );
1340 41 [GENM] := TRANS( [PHIB] ) * [ [MAA] * [PHIB] ];
1341 41 [GENK] := TRANS( [PHIB] ) * [ [KAA] * [PHIB] ];
1342 41 [DTSLP] := TRANS ( [BLSTJA] ) * [PHIB];
1343 41 [FTF] := TRANS ( [PHIB] ) * [BLGTJA];
1344 41 [GENF] := [BQDP] [FTF] * [BFRC];
1345 41 [GENFA] := [BQDP] [FTF] * [MATSS];
1346 41 [GENQ] := [GENFA] * [DTSLP];
1347 41 [GENQL] := [BQDP] [FTF] * [MATTR];
1348 41 CALL PARTN ( [GENQ], [QRR] , , [QRE], [QEE], [MPART] );
1349 41 CALL PARTN ( [GENK], , , , [KEE], [MPART] );
1350 41 [KEQE] := [QEE] + [KEE];
1351 41 CALL DECOMP ( [KEQE], [LKQ], [UKQ] );
1352 41 CALL ROWPART ( [GENF], [GFR], [GFE], [MPART] );
1353 41 CALL GPBS ( [LKQ], [UKQ], [GFE], [BTEM] );
1354 41 [DELM] := -(QRE) * [BTEM] + [GFR];
1355 41 CALL BLASTRIM ( BC, [DELM], [MRR(BC)], [URDB], [DELB] );
1356 41 [ELAS] := [BTEM] * [DELB];
1357 41 [SLPMOD] := TRANS ( [BLSTJA] ) * [PHIE];
1358 41 CALL BLASTDRV ( BC, [GENM], [GENK], [GENFA], [GENQL], [DELB],
1359 41 [URDB], [DWNWSH], [SLPMOD], [ELAS], [UBLASTI] );
1360 41 ENDIF;
1361 31$
1362 31$ BEGIN THE DATA RECOVERY OPERATIONS
1363 31$
1364 31 IF NBNDCOND > 1 CALL NULLMAT ( [UF], [AF], [PHIF] );
1365 31 IF NGDR <> 0 THEN
1366 41$ DATA RECOVERY WITH GDR
1367 41$
1368 41$
1369 41 IF BLOAD <> 0 THEN
1370 51 [UF] := [GSUBO(BC)] * [UA];
1371 51 IF NRSET <> 0 [AF] := [GSUBO(BC)] * [AA];
1372 51 ENDIF;
1373 41 IF BSAERO <> 0 AND TRMTYP > 0 THEN
1374 51 [UF] := [GSUBO(BC)] * [UA];
1375 51 IF NRSET <> 0 [AF] := [GSUBO(BC)] * [AA];
1376 51 ENDIF;
1377 41 IF BMODES <> 0 [PHIF] := [GSUBO(BC)] * [PHIA];
1378 41 IF BDTR <> 0 OR BMTR <> 0 [UTRANA] := [GSUBO(BC)] * [UTRANA];
1379 41 IF BDFR <> 0 OR BMFR <> 0 [UFREQA] := [GSUBO(BC)] * [UFREQA];
1380 41 ELSE

```

Figure C.1 Standard MAPOL Sequence (Continued)



\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
1381 4! IF NOMIT <> 0 THEN
1382 5!$
1383 5!$ DATA RECOVERY WITH STATIC CONDENSATION
1384 5!$
1385 5! IF BLOAD <> 0 THEN
1386 6! CALL RECOVA ( [UA], [PO], [GSUBO(BC)], NRSET, [AA],
1387 6! [IFM(BC)], , [KOOINV(BC)], [PFOA(BC)], [UF] );
1388 6! IF NRSET <> 0 CALL RECOVA ( [AA], , [GSUBO(BC)], , ,
1389 7! [PFOA(BC)], [AF] );
1390 6! ENDF;
1391 5! IF BSAERO <> 0 AND TRMTYP > 0 THEN
1392 6! CALL RECOVA ( [UA], [PO], [GASUBO(BC)], NRSET, [AA],
1393 6! [IFM(BC)], BSAERO, [KOOL(BC)], [KOOU(BC)],
1394 6! [PFOA(BC)], [UF] );
1395 6! IF NRSET <> 0 CALL RECOVA ( [AA], , [GASUBO(BC)], , ,
1396 7! [PFOA(BC)], [AF] );
1397 6! ENDF;
1398 5! IF BMODES <> 0 THEN
1399 6! [PHIO] := [GSUBO(BC)] * [PHIA];
1400 6! CALL ROWMERGE ( [PHIF], [PHIO], [PHIA], [PFOA(BC)] );
1401 6! ENDF;
1402 5! IF BDTR <> 0 OR BMTR <> 0 THEN
1403 6! CALL RECOVA ( [UTRANA], , [GSUBO(BC)], , ,
1404 6! [PFOA(BC)], [UTRANF] );
1405 6! ENDF;
1406 5! IF BDPR <> 0 OR BMFR <> 0 THEN
1407 6! CALL RECOVA ( [UFREQA], , [GSUBO(BC)], , ,
1408 6! [PFOA(BC)], [UFREQF] );
1409 6! ENDF;
1410 5! ELSE
1411 5!$
1412 5!$ DATA RECOVERY WITHOUT F-SET REDUCTION
1413 5!$
1414 5! IF BLOAD <> 0 THEN
1415 6! [UF] := [UA];
1416 6! IF NRSET <> 0 [AF] := [AA];
1417 6! ENDF;
1418 5! IF BSAERO <> 0 AND TRMTYP > 0 THEN
1419 6! [UF] := [UA];
1420 6! IF NRSET <> 0 [AF] := [AA];
1421 6! ENDF;
1422 5! IF BMODES <> 0 [PHIF] := [PHIA];
1423 5! IF BDTR <> 0 OR BMTR <> 0 [UTRANF] := [UTRANA];
1424 5! IF BDPR <> 0 OR BMFR <> 0 [UFREQF] := [UFREQA];
1425 5! ENDF;
1426 4! ENDF;
1427 3!$
1428 3! IF NBNDCOND > 1 CALL NULLMAT ( [UN], [AN], [PHIN] );
1429 3! IF NSPC <> 0 THEN
1430 4!$
1431 4!$ DATA RECOVERY WITH SPC-REDUCTION
1432 4!$
1433 4! IF BLOAD <> 0 THEN
1434 5! CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)] );
1435 5! IF NRSET <> 0 CALL YSMERGE ( [AN], , [AF], [PNSF(BC)] );
1436 5! ENDF;
1437 4! IF BSAERO <> 0 AND TRMTYP > 0 THEN
1438 5! CALL YSMERGE ( [UN], [YS(BC)], [UF], [PNSF(BC)] );
1439 5! IF NRSET <> 0 CALL YSMERGE ( [AN], , [AF], [PNSF(BC)] );
1440 5! ENDF;

```

Figure C.1 Standard MAPOL Sequence (Continued)

```

***** MAPOL SOURCE CODE LISTING *****

STAT  LEVEL
1441  41      IF BMODES <> 0 CALL YSMERGE ( [PHIN], [YS(BC)], [PHIF],
1442  51              [PNSF(BC)] );
1443  41      IF BDTR <> 0 OR BMTR <> 0
1444  51              CALL YSMERGE ( [UTRANN], [YS(BC)], [UTRANF],
1445  51              [PNSF(BC)], BDTR );
1446  41      IF BDFR <> 0 OR BMFR <> 0
1447  51              CALL YSMERGE ( [UFREQN], [YS(BC)], [UFREQF],
1448  51              [PNSF(BC)], BDFR );
1449  41      ELSE
1450  41$
1451  41$      DATA RECOVERY WITHOUT SPC-REDUCTION
1452  41$
1453  41      IF BLOAD <> 0 THEN
1454  51          [UN] := [UF];
1455  51          IF NRSET <> 0 [AN] := [AF];
1456  51      ENDIF;
1457  41      IF BSAERO <> 0 AND TRMTYP > 0 THEN
1458  51          [UN] := [UF];
1459  51          IF NRSET <> 0 [AN] := [AF];
1460  51      ENDIF;
1461  41      IF BMODES <> 0 [PHIN] := [PHIF];
1462  41      IF BDTR <> 0 OR BMTR <> 0 [UTRANN] := [UTRANA];
1463  41      IF BDFR <> 0 OR BMFR <> 0 [UFREQN] := [UFREQA];
1464  41      ENDIF;
1465  31$
1466  31      IF NENDCOND > 1 CALL NULLMAT ( [UG(BC)], [AG(BC)], [PHIG(BC)] );
1467  31      IF NMPC <> 0 THEN
1468  41$
1469  41$      DATA RECOVERY WITH MPC-REDUCTION
1470  41$
1471  41      IF BLOAD <> 0 THEN
1472  51          [UM] := [TMN(BC)] * [UN];
1473  51          CALL ROWMERGE ( [UG(BC)], [UM], [UN], [PGMN(BC)] );
1474  51          IF NRSET <> 0 THEN
1475  61              [UM] := [TMN(BC)] * [AN];
1476  61              CALL ROWMERGE ( [AG(BC)], [UM], [AN], [PGMN(BC)] );
1477  51          ENDIF;
1478  51      ENDIF;
1479  41      IF BSAERO <> 0 AND TRMTYP > 0 THEN
1480  51          [UM] := [TMN(BC)] * [UN];
1481  51          CALL ROWMERGE ( [UG(BC)], [UM], [UN], [PGMN(BC)] );
1482  51          IF NRSET <> 0 THEN
1483  61              [UM] := [TMN(BC)] * [AN];
1484  61              CALL ROWMERGE ( [AG(BC)], [UM], [AN], [PGMN(BC)] );
1485  61          ENDIF;
1486  51      ENDIF;
1487  41      IF BMODES <> 0 THEN
1488  51          [UM] := [TMN(BC)] * [PHIN];
1489  51          CALL ROWMERGE ( [PHIG(BC)], [UM], [PHIN], [PGMN(BC)] );
1490  51      ENDIF;
1491  41      IF BDTR <> 0 OR BMTR <> 0 THEN
1492  51          [UM] := [TMN(BC)] * [UTRANN];
1493  51          CALL ROWMERGE ( [UTRANG], [UM], [UTRANN], [PGMN(BC)] );
1494  51      ENDIF;
1495  41      IF BDFR <> 0 OR BMFR <> 0 THEN
1496  51          [UM] := [TMN(BC)] * [UFREQN];
1497  51          CALL ROWMERGE ( [UFREQG], [UM], [UFREQN], [PGMN(BC)] );
1498  51      ENDIF;
1499  41      ELSE

```

Figure C.1 Standard MAPOL Sequence (Continued)

\*\*\*\*\* MAPOL SOURCE CODE LISTING \*\*\*\*\*

```

STAT LEVL
1500 4!$                               $!
1501 4!$      DATA RECOVERY WITHOUT MPC-REDUCTION      $!
1502 4!$                               $!
1503 4!      IF BLOAD <> 0 THEN                               !
1504 5!          [UG(BC)] := [UN];                               !
1505 5!          IF NRSET <> 0 [AG(BC)] := [AN];               !
1506 5!      ENDIF;                                           !
1507 4!      IF BSAERO <> 0 AND TRMTYP > 0 THEN               !
1508 5!          [UG(BC)] := [UN];                               !
1509 5!          IF NRSET <> 0 [AG(BC)] := [AN];               !
1510 5!      ENDIF;                                           !
1511 4!      IF BMODES <> 0 [PHIG(BC)] := [PHIN];             !
1512 4!      IF BDTR <> 0 OR BMTR <> 0 [UTRANG] := [UTRANN];    !
1513 4!      IF BDPR <> 0 OR BMFR <> 0 [UFREQG] := [UFREQN];    !
1514 4!      ENDIF;                                           !
1515 3!$                               $!
1516 3!$      RECOVER PHYSICAL BLAST DISCIPLINE DISPLACEMENTS $!
1517 3!$                               $!
1518 3!      IF BBLAST <> 0 [UBLASTG] := [PHIG(BC)] * [UBLASTI]; !
1519 3!$                               $!
1520 3!$      HANDLE OUTPUT REQUESTS                          $!
1521 3!$                               $!
1522 3!      CALL OFFLOAD ( NUMOPTBC, BC, , GSIZE, [PG], TRMTYP, QDP, !
1523 3!          [GTKG], [AIRFC(MINDEX)], [DELTA] );             !
1524 3!      CALL OFFDISP ( NUMOPTBC, BC, NITER, GSIZE, [UG(BC)], [AG(BC)], !
1525 3!          TRMTYP, [UG(BC)], [AG(BC)], [UBLASTG], , [UTRANG], !
1526 3!          [UTRANE], [UFREQG], [UFREQE], LAMBDA, [PHIG(BC)] ); !
1527 3!      CALL EDR ( NUMOPTBC, BC, , NDV, GSIZE, EOSUMPRY, EODISC, !
1528 3!          [UG(BC)], [UG(BC)], , [UTRANG], [UFREQG], [PHIG(BC)] ); !
1529 3!      CALL OFFPEDR ( BC, HSIZE, , TRMTYP );              !
1530 3!      ENDDO;                                           !
1531 2!ENDIF;                                               !
1532 1!END;                                                 !

```

Figure C.1 Standard MAPOL Sequence (Concluded)

## APPENDIX D

### Solution Control Command Descriptions

This appendix contains the descriptions of the ASTROS Solution control commands. These commands are used in the Solution Control Packet to select the particular disciplines and analyses to be performed by the ASTROS procedure.

Solution Control Command : ANALYZE

Description: The first command in the ANALYZE subpacket

Hierarchy Level: Type of boundary condition

Format and Example(s):

ANALYZE

ANALYZE

**Solution Control Command : BLAST**

**Description:** Invokes the blast discipline

**Hierarchy Level:** Discipline

**Format and Example(s):**

BLAST Type (BLCOND - i, TSTEP - j)

BLAST MODAL (BLCOND - 20, TSTEP - 30)

**Option**

**Meaning**

Type

DIRECT or MODAL. Selects the solution approach.  
Default - MODAL.

i

Set identification of a BLAST bulk data entry used to specify nuclear blast parameters.

j

Set identification of TSTEP bulk data entries used to specify time step data for the blast response analysis.

**Remarks:**

1. BLAST does not generate design constraints for optimization.

Solution Control Command : BOUNDARY

Description: Specifies the displacement sets and related data to be used in a particular boundary condition.

Hierarchy Level: Boundary condition

Format and Example(s):

BOUNDARY MPC = i, SPC = j, REDUCE = k, SUPPORT = l, METHOD = m, K2PP = n,  
M2PP = o, B2PP = p, DYNRED = q, INERTIA = r, TFL = s, ESET = t,  
DAMPING = u

BOUNDARY SPC = 6

BOUNDARY SPC = 10, REDUCE = 20, SUPPORT = 30

BOUNDARY SPC = 10, METHOD = 5, DYNRED = 25, K2PP = STIFF, M2PP = MASS

Option

Meaning

i	Set identification of a multipoint constraint set. Invokes MPC and MPCADD bulk data entries (Integer > 0)
j	Set identification of a single-point constraint. Invokes SPC, SPC1 and SPCADD bulk data entries (Integer > 0)
k	Set identification of a static condensation set. Invokes ASET, ASET1, OMIT and OMIT1 bulk data entries (Integer > 0)
l	Set identification of the free body support. Invokes the SUPORT bulk data entry (Integer > 0)
m	Identifies the EIGR bulk data entry to be used (Integer > 0)
n	Selects the direct input stiffness matrix. Refers to a DMI or DMIG bulk data entry (Character)
o	Selects the direct input mass matrix. Refers to a DMI or DMIG bulk data entry (Character)
p	Selects the direct input damping matrix. Refers to a DMI or DMIG bulk data entry (Character)
q	Selects the dynamic reduction parameters from the DYNRED bulk data entry (Integer > 0)
r	Selects the JSETi bulk data entries to be used while performing dynamic reduction (Integer > 0)
s	Selects the transfer function set to be added to the input matrices. Refers to TF bulk data entries (Integer > 0)
t	Set identification of the extra degrees of freedom for the boundary condition. Invokes the EPOINT bulk data entries.
u	Set identification of damping data. Invokes TABDMP and VSDAMP bulk data entries.

- Remarks:**
1. Note that the REDUCE and ESET set specifications are an innovation relative to NASTRAN
  2. The bulk data entries will not be used in ASTROS unless selected in solution control.
  3. None of the options are required but at least one is needed.
  4. K2PP, M2PP, B2PP, TFL and DAMPING options are supported for dynamic disciplines only (i.e., for the FLUTTER, TRANSIENT and FREQUENCY disciplines).
  5. M2PP, B2PP and K2PP names will typically refer to DMI or DMIG entries, but may refer to any existing data base matrix entity of the proper dimension.



Solution Control Command : END

Description: Indicates the end of a subpacket.

Hierarchy Level: End

Format and Example(s):

END

END

Remarks: 1. ANALYZE and OPTIMIZE subpackets require their own END command.

Solution Control Command : FLUTTER

Description: Invokes the flutter analysis discipline

Hierarchy level: Discipline

Format and Example(s):

FLUTTER (FLCOND - i, DCONS - j)

FLUTTER

Option

Meaning

- |   |   |
|---|---|
| i | Set identification of a FLUTTER bulk data entry that provides flutter parameters.           |
| j | Set identification of DCONFLT bulk data entries which define flutter constraint conditions. |

Remarks: 1. The FLCOND parameter is required, DCONS is optional.

Solution Control Command : FREQUENCY

Description: Invokes the frequency analysis discipline

Hierarchy level: Discipline

Format and Example(s):

FREQUENCY Type (DLOAD - i, FSTEP - j, GUST - k)

FREQUENCY MODAL (DLOAD - 10, FSTEP - 20)

FREQUENCY DIRECT (DLOAD=100, FSTEP=30, GUST=55)

Option

Meaning

Type	DIRECT or MODAL. Selects the solution approach.
i	Set identification of a DLOAD bulk data entry.
j	Set identification of frequency bulk data entries (i.e., FREQ, FREQ1, FREQ2) that define the frequency steps of the analysis.
k	Set identification of a GUST bulk data entry which defines the gust parameters.

- Remarks:
1. The FREQUENCY discipline does not generate design constraints for optimization.
  2. Type, DLOAD and FSTEP are required.
  3. No more than one FREQUENCY analysis can be done in a single boundary condition.

**Solution Control Command : LABEL**

**Description:** Provides identifying information on subcase output.

**Hierarchy Level:** Label information

**Format and Example(s):**

**LABEL - n**

**LABEL - SYMMETRIC MANEUVER LOAD**

**Option**

**Meaning**

n

Any descriptive message that the user wishes to use to distinguish output.

- Remarks:**
1. LABEL information is used until it is superseded.
  2. The LABEL command is optional.
  3. Labels are limited to no more than 72 characters.

Solution Control Command : MODES

Description: Selects the Normal Modes discipline.

Hierarchy Level: Discipline

Format and Example(s):

MODES (DCONS - n)

MODES

MODES (DCONS - 10)

Option

Meaning

n

Set identification of DCONFRQ bulk data entries which define frequency constraints for the optimization task.

Remarks: 1. Only one modal analysis can be performed in a boundary condition using the EIGR data entry selected on the BOUNDARY command.

**Solution Control Command : OPTIMIZE**

**Description:** Invokes the ASTROS design capability

**Hierarchy Level:** Type of boundary condition

**Format and Example(s):**

**OPTIMIZE STRATEGY - yxxx**

**OPTIMIZE STRATEGY - 57**

**Option**

**Meaning**

yxxx

Selects the strategy to be used in the optimization procedure.

- Remarks:**
1. The STRATEGY parameter is a 4 digit number. In the example shown, the strategy is actually 0057 with the zeros defaulted.
  2. Any non-zero value of the first digit, "y", selects fully stressed design. The remaining 3 digits are not used at present, but are intended for use in directing other optimization methods.

Solution Control Command : PRINT

Description: Specifies the required output file processing for the print file

Hierarchy Level: Various

Format and Example(s):

PRINT (Form) PRES(Form) - a, VELO(Form) - b, DISP(Form) - c, ENER(Form) - d,  
FORC(Form) - e, GPFO(Form) - f, LOAD(Form) - g, SPCF(Form) - h,  
STRE(Form) - i, ACCE(Form) - j, STRA(Form) - k, ROOT - l,  
FREQ - m, MODE - n, TIME - o, DESIGN, DCONS, TRIM

PRINT DISP - ALL

PRINT (RECT) DISP - 6, ENERGY(POLA) - 10

PRINT MODES-NONE

Options

Meaning

Form	Meaning
	RECT or POLAR requests output in RECTangular or POLAR format (See Remark 1).
a	Set identification of an ELEMLIST bulk data entry that is used to request the aerodynamic panels at which pressures are to be printed.
b	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which velocities are to be printed.
c	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which displacements are to be printed.
d	Set identification of an ELEMLIST bulk data entry that is used to request the elements for which strain energies are to be printed.
e	Set identification of a ELEMLIST bulk data entry that is used to request the element for which forces are to be printed.
f	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which grid point forces are to be printed.
g	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which applied loads are to be printed.
h	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which SPC forces are to be printed.
i	Set identification of an ELEMLIST bulk data entry that is used to request the elements for which stresses are to be printed.
j	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which accelerations are to be printed.

- k** Set identification of an ELEMLIST bulk data entry that is used to request the elements at which strains are to be printed.
- l** Set identification of a MODELIST bulk data entry that is used to request the modes for which flutter root results are to be printed.
- m** Set identification of a FREQLIST bulk data entry that is used to request frequency output.
- n** Set identification of a MODELIST bulk data entry that is used to request the modes for which eigenvectors are to be printed.
- o** Set identification of a TIMELIST bulk data entry that is used to specify times of a transient response at which data is to be printed.

- REMARKS:**
1. Form is an optional parameter for printing complex data. RECTangular data outputs complex data with real and imaginary components while POLar outputs complex data using magnitude and phase. If used with the PRINT command, all complex data that are not otherwise specified use the requested Form. If used with another option, the Form overrides the global request.
  2. Options a through o can be either ALL, NONE or a positive integer. ALL requests all values. NONE turns off a request from a previous hierarchy while an integer value refers to a bulk data entry.
  3. DESIGN, DCONS and TRIM are toggles. If they are present, the specified data are printed. DESIGN indicates that information on the global and local design variables are to be printed at each iteration. DCONS indicates that constraint information is to be printed at each iteration. TRIM indicates that stability derivative data associated with an aeroelastic trim are to be printed.



Solution Control Command : PUNCH

Description: Specifies the required output file processing for the punch file

Hierarchy Level: Various

Format and Example(s):

PUNCH (Form) PRES(Form) = a, VELO(Form) = b, DISP(Form) = c, ENER(Form) = d,  
FORC(Form) = e, GPFO(Form) = f, LOAD(Form) = g, SPCF(Form) = h,  
STRE(Form) = i, ACCE(Form) = j, STRA(Form) = k, ROOT = l,  
FREQ = m, MODE = n, TIME = o, DESIGN, DCONS, TRIM

PUNCH DISP = ALL

PUNCH (RECT) DISP = 6, ENERGY(POLA) = 10

PUNCH MODES=NONE

Options

Meaning

Form	RECT or POLAR requests output in RECTangular or POLAR format (See Remark 1).
a	Set identification of an ELEMLIST bulk data entry that is used to request the aerodynamic panels at which pressures are to be punched.
b	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which velocities are to be punched.
c	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which displacements are to be punched.
d	Set identification of an ELEMLIST bulk data entry that is used to request the elements for which strain energies are to be punched.
e	Set identification of a ELEMLIST bulk data entry that is used to request the element for which forces are to be punched.
f	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which grid point forces are to be punched.
g	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which applied loads are to be punched.
h	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which SPC forces are to be punched.
i	Set identification of an ELEMLIST bulk data entry that is used to request the elements for which stresses are to be punched.
j	Set identification of a GRIDLIST bulk data entry that is used to request the grid points at which accelerations are to be punched.

- k Set identification of an ELEMLIST bulk data entry that is used to request the elements at which strains are to be punched.
- l Set identification of a MODELIST bulk data entry that is used to request the modes for which flutter root results are to be punched.
- m Set identification of a FREQLIST bulk data entry that is used to request frequency output.
- n Set identification of a MODELIST bulk data entry that is used to request the modes for which eigenvectors are to be punched.
- o Set identification of a TIMELIST bulk data entry that is used to specify times of a transient response at which data is to be punched.

- REMARKS:
1. Form is an optional parameter for punching complex data. RECTangular data outputs complex data with real and imaginary components while POLAR outputs complex data using magnitude and phase. If used with the PUNCH command, all complex data that are not otherwise specified use the requested Form. If used with another option, the Form overrides the global request.
  2. Options a through o can be either ALL, NONE or a positive integer. ALL requests all values. NONE turns off a request from a previous hierarchy while an integer value refers to a bulk data entry.
  3. DESIGN, DCONS and TRIM are toggles. If they are present, the specified data are punched. DESIGN indicates that information on the global and local design variables are to be punched at each iteration. DCONS indicates that constraint information is to be punched at each iteration. TRIM indicates that stability derivative data associated with an aeroelastic trim are to be punched.

Solution Control Command : SAERO

Description: Invokes the static aeroelastic discipline.

Hierarchy Level: Discipline

Format and Example(s):

SAERO (TRIM - m, DCONS - n)

SAERO (TRIM - 60)

SAERO (TRIM - 70, DCONS - 10)

Option

Meaning

m

Set identification of a TRIM bulk data entry which provides flight configuration information.

n

Set identification of DCONDSP, DCONALE and DCONCLA bulk data entries which define constraints in an optimization task for a given trim condition.

- Remarks:
1. TRIM is required, DCONS is optional.
  2. If SAERO disciplines are selected in the Solution Control, only other SAERO disciplines can appear in the same boundary condition.

**Solution Control Command : SOLUTION**

**Description:** The first command in the solution control packet.

**Hierarchy Level:** Beginning of solution

**Format and Example(s):**

**SOLUTION**

**SOLUTION**

**Remarks:** 1. One and only one SOLUTION command must always appear as the first command of the solution control packet.

Solution Control Command : STATICS

Description: Invokes the statics analysis discipline

Hierarchy level: Discipline

Format and Example(s):

STATICS (MECH - i, THERMAL - j, GRAVITY - k, DCONS - l)

STATICS (MECH - 10)

STATICS (MECH - 4, THERMAL - 6, DCONS - 12)

Option

Meaning

i	Set identification for external loads as defined by LOAD, PLOAD, FORCE, FORCE1, MOMENT, and MOMENT1 bulk data entries.
j	Set identification for temperatures as defined by TEMP or TEMPD bulk data entries.
k	Set identification of GRAV bulk data entries which define gravity forces.
l	Set identification of DCONDSP bulk data entries which define displacement constraints in an optimization task with a statically applied load.

- Remarks.
1. The sum of all the loads forms a single right hand side for a statics analysis.
  2. At least one of the load types must be present. The DCONS parameter is optional.
  3. Gravity forces may be included indirectly if referenced by the LOAD bulk data entry.

**Solution Control Command : SUBTITLE**

**Description:** Defines a subtitle which will appear in the output.

**Hierarchy Level:** Label information

**Format and Example(s):**

SUBTITLE - n

SUBTITLE - SUPERSONIC DESIGN CONDITION

**Option**

**Meaning**

n

Any descriptive information can be inserted here

- Remarks:
1. SUBTITLE information is used until it is superseded.
  2. The SUBTITLE command is optional.
  3. Subtitles are limited to 72 characters.

Solution Control Command : TITLE

Description: Defines a title which will appear in the output.

Hierarchy Level Label information

Format and Example(s):

TITLE - n

TITLE - DESIGN OF A FORWARD SWEPT WING MODEL

Option

Meaning

n

Any descriptive information can be inserted here

- Remarks:
1. TITLE information is used until it is superseded.
  2. The TITLE command is optional.
  3. Titles are limited to no more than 72 characters.

**Solution Control Command : TRANSIENT**

**Description:** Invokes the transient response analysis

**Hierarchy Level:** Discipline

**Format and Example(s):**

TRANSIENT Type (DLOAD - i, TSTEP - j, FFT - k, IC - l, GUST - m)

TRANSIENT MODAL (DLOAD - 10, TSTEP - 20)

TRANSIENT DIRECT (DLOAD=100, TSTEP=30, IC = 45)

TRANSIENT MODAL (DLOAD=35, TSTEP=2, FFT=999, GUST=45)

**Option**

**Meaning**

Type	DIRECT or MODAL. Selects the solution approach.
i	Set identification of a DLOAD bulk data entry.
j	Set identification of TSTEP bulk data entries which provide time step information for the analysis.
k	Set identification of a FFT bulk data entry which provides parameters to use Fast Fourier Transform methods in performing the transient analysis
l	Set identification of IC bulk data entries which define initial conditions.
m	Set identification of a GUST bulk data entry which defines the gust parameters.

- Remarks:**
1. The TRANSIENT discipline does not generate design constraints for optimization.
  2. Type, DLOAD and TSTEP are required.
  3. If GUST is present, FFT must also be used.
  4. IC cannot be used with the MODAL solution method. IC also cannot be used with FFT or GUST.
  5. No more than one TRANSIENT response analysis can be performed in a single boundary condition.



## APPENDIX E

### BULK DATA ENTRIES

This appendix contains the complete documentation for all bulk data packet inputs to the ASTROS system. Each field of each bulk data entry is described in detail regarding its use in the ASTROS system, the data type requirements, and any defaults that are defined. In addition, the bulk data descriptions include documentation regarding interactions both among bulk data entries and among the data fields of each data entry. These data are listed in alphabetical order by bulk data entry name.

Input Data Entry    \$Comment

Description:    For user convenience in inserting commentary material into the unsorted echo of the input Bulk Data Deck. The \$ entry is otherwise ignored by the program. These entries will not appear in a sorted echo.

Format and Example:

1	2	3	4	5	6	7	8	9	10
\$	Followed by any legimate characters in columns 2-80								
	THIS IS A REMARK (*."\$\$)--/								

**Input Data Entry    AEFACT    Aerodynamic Lists**

**Description:**    Used to specify lists of real numbers for aeroelastic analysis.

**Format and Example:**

1	2	3	4	5	6	7	8	9	10
<u>AEFACT</u>	<u>SID</u>	<u>D1</u>	<u>D2</u>	<u>D3</u>	<u>D4</u>	<u>D5</u>	<u>D6</u>	<u>D7</u>	<u>CONT</u>
AEFACT	97	.3	.7	1.0					
<u>CONT</u>	<u>D8</u>	<u>D9</u>	<u>-ETC-</u>						

**Field**

**Contents**

**SID**                      Set identification number (Unique Integer > 0).

**Di**                        Number (Real).

**Remarks:**

1. These factors are selected by AIRFOIL, AXSTA, CAEROi and/or PAEROi data entries.
2. Imbedded blank fields are forbidden.
3. If used to specify division points, note that there is one more division point than the number of divisions.

Input Data Entry    AERO            Aerodynamic Physical Data

Description:    Gives basic aerodynamic parameters for unsteady aerodynamic disciplines.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
AERO	ACSID	REFC	RHOREF						
AERO	100	300.0	1.1E-7						

Field

Contents

ACSID                    Aerodynamic coordinate system identification (Integer  $\geq 0$ ).  
See Remark 2.

REFC                    Reference length (for reduced frequency) (Real).

RHOREF                Reference density (Real).

Remarks:

1. This entry is required for unsteady aerodynamic disciplines. Only one AERO entry is allowed.
2. The ACSID must be a rectangular coordinate system. Flow is in the positive x-direction.

Input Data Entry AEROS Static Aero Physical Data

Description: Gives basic parameters for static aeroelasticity.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
AEROS	ACSID	RCSID	REFC	REFB	REFS	REFG	REFD	REFL	
AEROS	10	20	10.	100.	1000.	1			

Field

Contents

ACSID	Aerodynamic coordinate system identification (Integer $\geq 0$ ). See Remark 2.
RCSID	Reference coordinate system identification for rigid body motions (Integer $\geq 0$ ).
REFC	Reference chord length (Real $> 0.0$ )(D = 1.0)
REFB	Reference span (Real $> 0.0$ )(D = 1.0)
REFS	Reference wing area (Real $> 0.0$ )(D = 1.0)
REFG	Reference grid point for stability derivative calculations.
REFD	Fuselage reference diameter (Real $> 0$ )(D = 1.0)
REFL	Fuselage reference length (Real $> 0$ )(D = 1.0)

Remarks:

1. This entry is required for static aeroelasticity problems. Only one AEROS entry is allowed.
2. The ACSID must be a rectangular coordinate system. Flow is in the positive x-direction.
3. The RCSID must be a rectangular coordinate system. All degrees of freedom defining trim variables will be defined in this coordinate system.

Input Data Entry    AESURF       Aerodynamic Control Surface

Description:    Specifies an aerodynamic control surface.

Format and Example:

1	2	3	4	5	6	7	8	9	10
AESURF	SETID	LABEL	ACID1	CID1	FBOXID1	LBOXID1			CONT
AESURF	600	ELEV	6000	1	6010	6030			+BC
CONT			ACID2	CID2	FBOXID2	LBOXID2			
+BC			8000	2	8010	8030			

Field

Contents

SETID                    Identification number of an aerodynamic trim variable degree of freedom (Integer > 0).

LABEL                   An alphanumeric string of up to eight characters to identify the control surface.

ACID1                   ID of the aircraft component that the surface is on.

CID1                    Identification number of a rectangular coordinate system whose y-axis defines the hinge line of the control surface.

FBOXID1                First aero box on the control surface relative to ACID1.

LBOXID1                Last aero box on control surface, relative to ACID1.

Remarks:

1. Either one or two control surfaces may be defined, depending on the symmetry flag.
2. Allowable values of LABEL are: ELEV, AILERON and RUDDER.

## Input Data Entry AIRFOIL

**Description:** Defines airfoil properties for USSAERO.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
AIRFOIL	ACID	CMPNT	CP	ICHORD	IUST	ILST	ICAM	RADIUS	CONT
AIRFOIL	1	WING	1	10	20		30		abc
CONT	X1	Y1	Z1	X12	IPANEL				
+BC	0.0	0.0	0.0	50.					

### Field

### Contents

ACID	Associated aircraft component ID (Integer > 0).
CMPNT	Type of aircraft component (Text).
CP	Coordinate system for airfoil (Integer).
ICHORD	ID of an AEFACT data entry containing a list of division points (in terms of percent chord) at which airfoil data are specified (Integer).
IUST, ILST	ID of an AEFACT data entry containing a list of airfoil half thicknesses in percent chord at the chordwise cuts for the upper and lower surfaces, respectively (Integer).
ICAM	ID of an AEFACT data entry containing a list of airfoil camber values (z-ordinates expressed in percent chord) at the chordwise cuts (Integer).
RADIUS	Radius of the leading edge, expressed in percent chord (Real).
X1, Y1, Z1	Location of airfoil leading edge in coordinate system CP (Real).
X12	Airfoil chord length in coordinate system CP. (Real > 0).
IPANEL	ID of an AEFACT data entry containing a list of chordwise cuts for wing panelling.

### Remark:

1. Allowable components are WING, FIN and CANARD.
2. ILST and ICAM present redundant information so that, at most, only one can be non-zero.
3. ICAM cannot be defined for FIN and CANARD components. ILST cannot be defined for FIN components.

4. If the RADIUS field is blank, a round leading edge of radius zero is used.
5. IPANEL is optional and is used when different chordwise cuts on each end of the panel are desired.



**Input Data Entry    ASET    Selected Coordinates for the a-set**

**Description:** Defines degrees of freedom that the user desires to place in the analysis set. Used to define the number of independent degrees of freedom.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
<u>ASET</u>	<u>SETID</u>	<u>ID</u>	<u>C</u>	<u>ID</u>	<u>C</u>	<u>ID</u>	<u>C</u>		
ASET	16	2	23	3516					

**Field**

**Contents**

**SETID**                      The set identification number of the REDUCE set.

**ID**                              Grid or scalar point identification number (Integer > 0)

**C**                                Component number, zero or blank for scalar points, any unique combinations of the digits 1-6 for grid points.

**Remarks:**

1. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
2. When only ASET and/or ASET1 entries are present, all degrees of freedom not otherwise constrained will be placed in the o-set.
3. ASET entries must be selected in Solution Control (REDUCE-SETID) to be used

Input Data Entry    ASET1    Selected Coordinates for the a-set, Alternate Form

Description:    Defines degrees of freedom that the user desires to place in the analysis set.    Used to define the number of independent degrees of freedom.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
ASET1	SETID	C	G	G	G	G	G	G	CONT
ASET1	345	2	1	3	10	9	6	15	ABC
CONT	G	G	G	etc					
+BC	7	8		etc					

Alternate Form:

1	2	3	4	5	6	7	8	9	10
ASET1	SETID	C	ID1	"THRU"	ID2				
SET1	10	123456	7	THRU	109				

Field

Contents

SETID                      The REDUCE set identification number

C                            Component number (any unique combination of the digits 1-6 with no imbedded blanks) when point identification numbers are grid points; must be null or zero if point identification numbers are scalar points.

G, ID1, ID2                Grid or scalar point identification numbers (Integer > 0, ID2 > ID1)

Remarks:

1. Coordinates specified on this entry form members of a set that is exclusive from other sets defined by bulk data entries.
2. When only ASET and/or ASET1 entries are present, all degrees of freedom not otherwise constrained will be placed in the o-set.
3. If the alternate form is used, all points in the sequence ID1 through ID2 are required to exist.
4. ASET1 entries must be selected in Solution Control (REDUCE-SETID) to be used.

**Input Data Entry    ATTACH**

**Description:** Defines the aerodynamic control points to be attached to a reference grid for load transfer.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
<u>ATTACH</u>	<u>EID</u>	<u>MACROID</u>	<u>BOX1</u>	<u>BOX2</u>	<u>RGRID</u>				
ATTACH	100	111	111	118	1				

**Field**

**Contents**

**EID**                      Element identification number (Integer > 0)

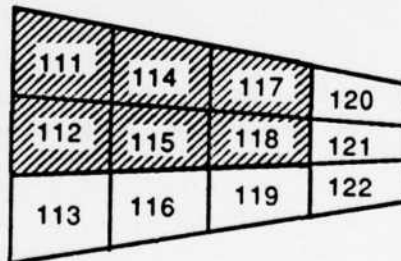
**MACROID**                Element id of a CAERO; or PAERO; element which contains the specified aerodynamic control points (Integer > 0)

**BOX1, BOX2**            Starting and final box whose force is to be transferred to the referenced grid (Integer > 0, BOX2 > BOX1)

**RGRID**                    Grid point id of reference grid point (Integer > 0)

**Remarks:**

1. The EID is used only for error messages.
2. This entry applies to both the steady and unsteady aerodynamic models.
3. The attached aerodynamic boxes are selected as shown below:



Input Data Entry AXSTA

Description: Defines body axial station parameters.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
AXSTA	BCID	XSTA	CBOD	ABOD	LYRAD	LZRAD			
AXSTA	10	10.00	0.5		10	20			

Field

Contents

BCID	Body component ID (Integer > 0)
XSTA	Value of the x-ordinate of the body station (Real)
CBOD	Value of the z-ordinate of the center line at this station. This defines the body camber (Real).
ABOD	Cross sectional area of the body at this station (Real).
LYRAD; LZRAD	ID of an AEFACT data entry containing a list of the y-ordinates (z-ordinates) of the body section. (Integer)

Remarks:

1. If ABOD is present, the body is assumed to be circular and the radial ordinates are computed at NRAD (cf. the BODY bulk data entry) equal intervals. No LYRAD and LZRAD data are allowed when ABOD is present.
2. If ABOD is blank, LYRAD and LZRAD data must be present.
3. For Pods, CBOD, LYRAD and LZRAD data are not permitted.
4. For the fuselage, XSTA is actual x location; for pods, XSTA is relative to the XLOC value given on the BODY bulk data entry.

Input Data Entry BAROR Simple Beam (BAR) Orientation Default Values

Description: Defines default values for fields 3 and 6 - 8 of the CBAR entry.

Format and Example:

1	2	3	4	5	6	7	8	9	10
BAROR	PID				X1,GO	X2	X3		
BAROR	39				0.6	2.9	-5.87		

Field

Contents

PID Identification number of PBAR property entry (Integer > 0 or blank)

X1,X2,X3 Vector components measured in displacement coordinate system at GA to determine (with the vector from end A to end B) the orientation of the element coordinate system for the bar element (Real or blank).

GO Grid point identification number (Integer > 0).

Remarks

1. The contents of fields on this entry will be assumed for any CBAR entry whose corresponding fields are blank.
2. Only one BAROR entry may appear in the user's Bulk Data Packet.

Input Data Entry BLAST Nuclear Blast Parameters

Description: Defines basic parameters needed for nuclear blast response analysis.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
BLAST	BLID	ALTITUDE	VELOCITY	WKT	BALT	MACH	DELTAX	DELTAY	CONT
BLAST	100	10000.	1000.0	1.E5	15000.	0.8			ABC
CONT	KGRD	HGRD	SYMxz	SYMxy	TRSUF	NZ			CONT
+BC			2	10	6.0				
CONT	TMIN	TMAX	NTIME	BMIN	BMAX	NBETA			

Field

Contents

BLID	Set identification number referenced by Solution Control. (Integer > 0 )
ALTITUDE	Aircraft altitude ( Real )
VELOCITY	Aircraft velocity (Real $\geq$ 0.0 )
WKT	Weapon yield ( Real )
BALT	Blast altitude ( Real )
MACH	Mach number ( Real $0.0 \leq$ Mach $\leq$ 1.0 )
DELTAX	X distance from aircraft to blast point ( Real )
DELTAY	Y distance from aircraft to blast point ( Real )
KGRD	Key denoting presence of the ground = 0, no ground = 1 include the ground
HGRD	Height of ground level ( Real )
SYMxz	Symmetry flag for blast analysis about xz plane
SYMxy	Symmetry flag for blast analysis about xy plane
TRSUF	ID of an AESURF entry used as the trim surface
NZ	Load factor for the trim calculation
TMIN	Minimum time used in the definition of polynomial curve fitting ( Real > 0.0, def = 0.10 )

<b>TMAX</b>	Maximum time used in the definition of polynomial curve fitting ( Real > 0.0, def = 20.0 )
<b>NTIME</b>	Number of time steps ( Integer > 0, def = 20 )
<b>BMIN</b>	Minimum Beta value used in the definition of polynomial curve fitting ( Real > 0.0, def = 0.375 )
<b>BMAX</b>	Maximum Beta value used in the definition of polynomial curve fitting ( Real > 0.0, def = 10.0 )
<b>NBETA</b>	Number of Beta values ( Integer > 0, def = 7 )

**Remarks:**

1. In order to be used, the BLID must be referenced by Solution Control.
2. The second continuation entry is not required.
3. The default values of TMIN, TMAX, NTIME, BMIN, BMAX and NBETA should be adequate.
4. The symmetry flags refer to the options selected on the MKAERO1 entries.

Input Data Entry BODY

Description: Defines body configuration parameters for steady aeroelasticity.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
BODY	BCID	CMPNT	CP	NRAD	XLOC	YLOC	ZLOC		
BODY	10	FUSEL	0	3					

Field

Contents

BCID                      Body component ID (Integer > 0)

CMPNT                    Component type (FUSEL for the fuselage or POD for a pod)

CP                        Coordinate system for the input geometry.

NRAD                    Number of equal radial cuts used to define the body.

XLOC;YLOC;  
ZLOC                    Ordinates of the body in the CP coordinate system

Remarks:

1. NRAD is input if equally spaced radial cuts are desired. Arbitrary radial cuts are specified using the AXSTA and AEFACT data entries.
2. The geometry given with the XLOC, YLOC, ZLOC entries is used only with POD components.



Input Data Entry CAERO1 Unsteady Aerodynamic Panel Element Connection

**Description:** Defines an aerodynamic macroelement (panel) in terms of two leading edge locations and side chords. This is used for Doublet-Lattice theory.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CAERO1	EID	PID	CP	NSPAN	NCHORD	LSPAN	LCHORD	IGID	CONT
CAERO1	1000	1		3			2	1	ABC
+BC	X1	Y1	Z1	X12	X4	Y4	Z4	X43	
+BC	0.0								

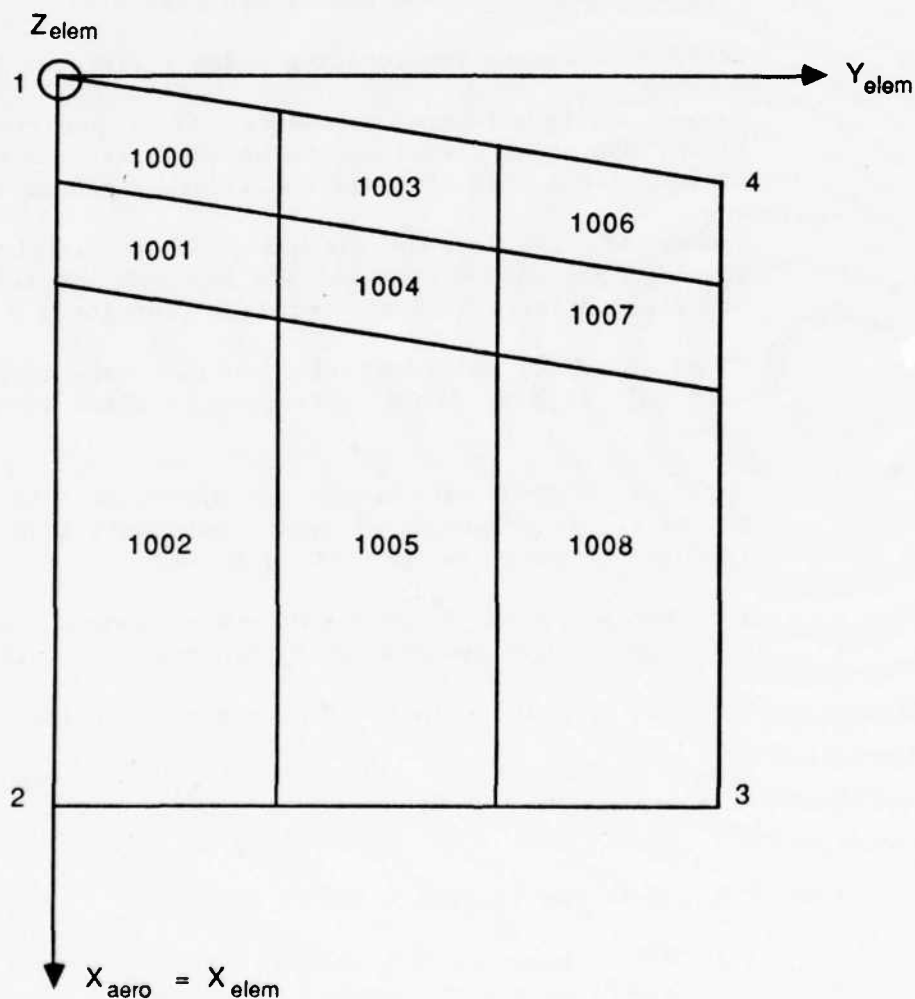
**Field**

**Contents**

EID	Element identification number (Integer > 0).
PID	Identification number of property entry (Integer $\geq 0$ or blank). Used to specify associated bodies.
CP	Coordinate system for locating points 1 and 4 (Integer > 0 or blank).
NSPAN	Number of spanwise boxes; if a positive value is given, NSPAN equal divisions are assumed; if zero or blank, a list of division points is given at LSPAN, field 7 (Integer $\geq 0$ or blank).
NCHORD	Number of chordwise boxes; if a positive value is given, NCHORD equal divisions are assumed; if zero or blank, a list of division points is given at LCHORD, field 8 (Integer $\geq 0$ or blank).
LSPAN	ID of an AEFACT data entry containing a list of division points for spanwise boxes. Used only if NSPAN, field 5, is zero or blank (Integer > 0 or blank).
LCHORD	ID of an AEFACT data entry containing a list of division points for chordwise boxes. Used only if NCHORD, field 6, is zero or blank (Integer > 0 or blank).
IGID	Interference group identification (aerodynamic elements with different IGID's are uncoupled) (Integer > 0).
X1,Y1,Z1; X4,Y4,Z4	Location of points 1 and 4, in coordinate system CP (Real).
X12; X43	Edge chord lengths (in aerodynamic coordinate system) (Real $\geq 0$ , and not both zero).

Remarks:

1. The boxes are numbered sequentially, beginning with EID.
2. The number of division points is one greater than the number of boxes. Thus, if  $NSPAN = 3$ , the division points are 0.0, 0.333, 0.667, 1.000. If the user supplies division points, the first and last points need not be 0. and 1. (in which case the corners of the panel would not be at the reference points).
3. A triangular element is formed if  $X_{12}$  or  $X_{43} = 0.0$
4. The element coordinate system (right-handed) is shown in the sketch below.
5. The continuation entry is required.



# Input Data Entry CAERO2 Unsteady Aerodynamic Body Connection

**Description:** Defines an aerodynamic body for Doublet-Lattice aerodynamics.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
CAERO2	EID	PID	CP	NSB	NINT	LSB	LINT	IGID	CONT
CAERO2	1500	2	100		4	99		1	abc
CONT	X1	Y1	Z1	X12					
+bc	-1.0	100	-30	175					

## Field

## Contents

EID	Element identification number (Integer > 0).
PID	Property identification number (Integer > 0).
CP	Coordinate system for locating point 1 (Integer ≥ 0).
NSB	Number of interference elements; if a positive number is given, NSB equal divisions are assumed; if zero or blank, see field 7 for a list of divisions (Integer ≥ 0 or blank).
NINT	Number of interference elements; if a positive number is given, NINT equal divisions are assumed; if zero or blank, see field 8 for a list of divisions (Integer ≥ 0 or blank).
LSB	ID of an AEFACT data entry for slender body division points; used only if NSB, field 5, is zero or blank (Integer ≥ 0 or blank).
LINT	ID of an AEFACT data entry containing a list of division points for interference elements; used only if NINT, field 6, is zero or blank (Integer > 0 or blank).
IGID	Interference group identification (aerodynamic elements with different IGID's are uncoupled) (Integer > 0).
X1,Y1,Z1	Location of point 1 in coordinate system CP (Real).
X12	Length of body in the x-direction of the aerodynamic coordinate system (Real > 0).

## Remarks:

1. Point 1 is the leading point of the body.
2. All CAERO1 (panels) and CAERO2 (bodies) in the same group (IGID) will have aerodynamic interaction.
3. At least one interference element is required for each aerodynamic body specified by this entry.

4. Element identification numbers on the aerodynamic bodies must have the following sequence:

- (A) Panels first
- (B) Z bodies (see PAERO2 orientation flag)
- (C) ZY bodies
- (D) Y bodies

### Input Data Entry CAERO6

**Description:** Defines an aerodynamic macroelement (panel) for steady aeroelasticity.

#### Format and Examples:

1	2	3	4	5	6	7	8	9	10
CAERO6	ACID	CMPNT	CP	IGRP	LCHORD	LSPAN			
CAERO6	1	2ING		1	20	30			

#### Field

#### Contents

ACID	Component ID (Integer > 0)
CMPNT	Aircraft component (Text)
CP	Coordinate system (Integer)
IGRP	Group number for this component (Integer)
LCHORD	ID of AEFACT data entries containing a list of division points in percent chord for chordwise boxes for aerodynamic surface. If LCHORD is zero, the chordwise divisions are identified by the IPANEL entry on the AIRFOIL bulk data entry (Integer $\geq 0$ , or blank).
LSPAN	ID of an AEFACT data entry containing a list of division points in terms of dimensional span stations for spanwise boxes. If this is zero or blank, the y locations from the AIRFOIL bulk data entries for the component ACID are used (Integer $\geq 0$ , or blank).

#### Remarks:

1. Allowable components are WING, FIN and CANARD.
2. The IGRP field allows related components to be processed together for interference effects; e.g., one group could be a wing/body/tail combination while a second group would be a pod/fin combination.
3. Note that chordwise cuts are in percent while spanwise cuts require physical coordinates. For spanwise cuts, y-coordinates are input for wings and canards while z-coordinates are input for fins.

# Input Data Entry CBAR Simple Beam Element Connection

Description: Defines a simple beam element (BAR) of the structural model.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
CBAR	EID	PID	GA	GB	X1,GO	X2	X3	TMAX	CONT
CBAR	2	39	7	3	13				123
CONT	PA	PB	W1A	W2A	W3A	W1B	W2B	W3B	
+23		513							

## Field

## Contents

EID	Unique element identification number (Integer > 0).
PID	Identification number of a PBAR property entry (Default is EID unless BAROR entry has nonzero entry in field 3) (Integer > 0)
GA,GB	Grid point identification numbers of connection points (Integer > 0).
X1,X2,X3	Components of vector (v), at end A, measured at end A, parallel to the components of the displacement coordinate system for GA, to determine (with the vector from end A to end B) the orientation of the element coordinate system for the BAR element (Real)
GO	Grid point identification number to optionally supply X1, X2, X3 (Integer > 0). Direction of orientation vector is GA to GO
TMAX	Maximum allowable cross-sectional area in design (Real > 0.0 or blank).
PA,PB	Pin flags for bar ends A and B, respectively (up to 5 of the unique digits 1-6 anywhere in the fields with no imbedded blanks; Integer ≥ 0). Used to remove connections between the grid point and selected degrees of freedom of the bar. The degrees of freedom are defined in the <u>element's</u> coordinate system. The bar must have stiffness associated with the pin flag. For example, if PA=4 is specified, the PBAR entry must have a value for J, the torsional stiffness.
W1A,W2A,W3A	Components of offset vectors (wa) and (wb), respectively, in displacement coordinate systems at points GA and GB, respectively (Real or blank).

## Remarks:

1. If there are no pin flags or offsets, the continuation entry may be omitted.

2. The TMAX value is used only for shape function design variable linking.
3. See the BAROR entry for default options for fields 3 and 6-8.

Input Data Entry CELAS1 Scalar Spring Connection

Description: Defines a scalar spring element of the structural model.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CELAS1	EID	PID	G1	C1	G2	C2	TMAX		
CELAS1	2	6			8	1			

Field

Contents

EID	Element identification number (Integer > 0)
PID	Identification number of a PELAS property entry (Default is EID) (Integer > 0)
G1, G2	Geometric grid point identification number (Integer $\geq$ 0)
C1, C2	Component number ( $6 \geq$ Integer $\geq$ 0)
TMAX	Maximum spring constant value for design

Remarks:

1. Scalar points may be used for G1 and/or G2, in which case the corresponding C1 and/or C2 must be zero or blank. Zero or blank may be used to indicate a grounded terminal G1 or G2 with a corresponding blank or zero C1 or C2. A grounded terminal is a point whose displacement is constrained to zero.
2. The two connection points (G1, C1) and (G2, C2) must be distinct.
3. TMAX is ignored unless the element is designed using shape function linking.



# Input Data Entry    CELAS2    Scalar Spring Property and Connection

**Description:**    Defines a scalar spring element of the structural model without reference to a property entry.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
CELAS2	EID	K	G1	C1	G2	C2	GE	S	CONT
CELAS2	28	6.2+3	32		19	4			
CONT	TMIN	TMAX							

## Field

## Contents

EID	Element identification number (Integer > 0).
K	The value of the scalar spring constant (Real).
G1, G2	Geometric grid point identification number (Integer ≥ 0).
C1, C2	Component number (6 ≥ Integer ≥ 0).
GE	Damping coefficient (Real).
S	Stress coefficient (Real).
TMIN, TMAX	Minimum and maximum spring constant values for design.

## Remarks:

- Scalar points may be used for G1 and/or G2 in which case the corresponding C1 and/or C2 must be zero or blank. Zero or blank may be used to indicate a grounded terminal G1 or G2 with a corresponding blank or zero C1 or C2. A grounded terminal is a point whose displacement is constrained to zero.
- This single entry completely defines the element since no material or geometric properties are required.
- The two connection points (G1, C1) and (G2, C2) must be distinct.
- The TMIN and TMAX values are ignored unless shape function design variable linking is used.

Input Data Entry    CIHEX1    Linear Isoparametric Hexahedron Element  
Connection

Description:    Defines a linear isoparametric hexahedron element of the structural model.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CIHEX1	EID	PID	G1	G2	G3	G4	G5	G6	CONT
CIHEX1	137	5	3	8	5	4	9	14	ABC
CONT	G7	G8							
+BC	27	35							

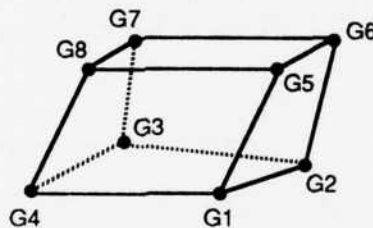
Field

Contents

EID                      Element identification number (Integer > 0).

PID                      Identification number of a PIHEX property entry (Integer > 0).

G1,...,G8              Grid point identification numbers of connection points (Integer > 0, G1 ≠ G2 ≠ ...G8).



Remarks:

1. Grid points G1, G2, G3, and G4 must be given in counter-clockwise order about one quadrilateral, with G1 and G5 along the same edge.
2. There is no non-structural mass.
3. The quadrilateral faces need not be planar.
4. Stresses are given in the basic coordinate system.
5. The continuation is required.
6. No physical property of this element can be used as a local design variable for automated design.

Input Data Entry    CIHEX2    Quadratic Isoparametric Hexahedron Element  
Connection

**Description:** Defines a quadratic isoparametric hexahedron element of the structural model.

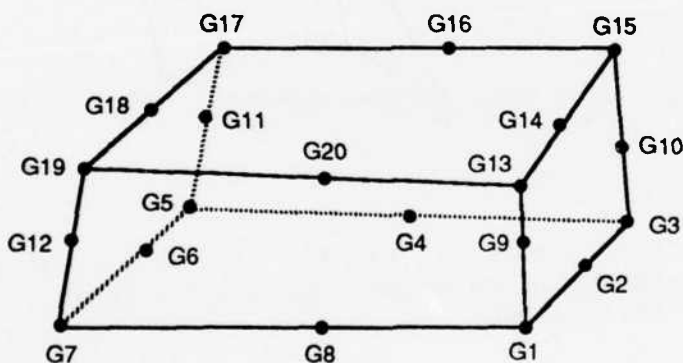
**Format and Example:**

1	2	3	4	5	6	7	8	9	10
CIHEX1	EID	PID	G1	G2	G3	G4	G5	G6	CONT
CIHEX1	110	7	3	8	12	13	14	9	ABC
CONT	G7	G8	G9	G10	G11	G12	G13	G14	CONT
+BC	5	4	16	19	20	17	23	27	DEF
CONT	G15	G16	G17	G18	G19	G20			
+EF	31	32	33	28	25	24			

**Field**

**Contents**

- EID                      Element identification number (Integer > 0).
- PID                      Identification number of a PIHEX property entry (Integer > 0).
- G1,...,G20              Grid point identification numbers of connection points (Integer > 0, G1 ≠ G2 ≠ .... ≠ G20).



**Remarks:**

1. Grid points G1,...,8 must be given in counter-clockwise order about one quadrilateral face when viewed from inside the element. G9,...,G12 and G13,...,G20 are in the same direction with G1, G9 and G13 along the same edge.
2. There is no non-structural mass.
3. The quadrilateral faces need not be planar.
4. Stresses are given in the basic coordinate system.

5. Both continuations are required.
6. No physical property of this element can be used as a local design variable in automated design.

Input Data Entry    CIHEX3    Cubic Isoparametric Hexahedron  
Element Connection

**Description:** Defines a cubic isoparametric hexahedron element of the structural model.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CIHEX1	EID	PID	G1	G2	G3	G4	G5	G6	CONT
CIHEX1	15	3	4	9	12	17	18	19	ABC
CONT	G7	G8	G9	G10	G11	G12	G13	G14	CONT
+BC	20	13	10	7	6	5	22	25	DEF
CONT	G15	G16	G17	G18	G19	G20	G21	G22	CONT
+EF	26	23	28	31	32	29	36	41	GHI
CONT	G23	G24	G25	G26	G27	G28	G29	G30	CONT
+HI	106	213	413	95	67	40	45	90	+KL
CONT	G31	G32							
+KL	38	37							

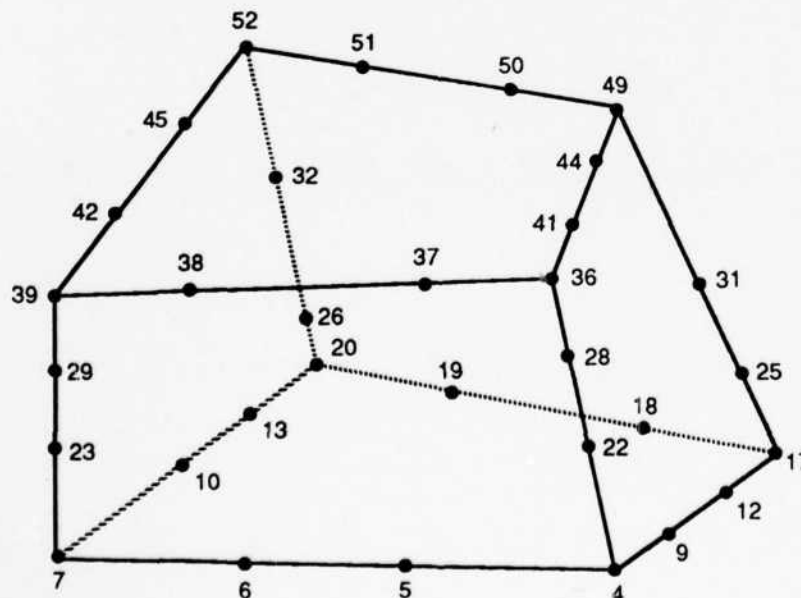
**Field**

**Contents**

EID                    Element identification number (Integer > 0).

PID                    Identification number of a PIHEX property entry (Integer > 0)

G1,...,G32            Grid point identification number of connection points (Integer > 0, G1 ≠ G2 ≠ ... ≠ G32).



Remarks:

1. Grid points  $G_1, \dots, G_{12}$  must be given in counter-clockwise order about one quadrilateral face when viewed from inside the element.  $G_{13}, \dots, G_{16}$ ;  $G_{17}, \dots, G_{20}$ ; and  $G_{21}, \dots, G_{32}$  are in the same direction with  $G_1$ ,  $G_{13}$ ,  $G_{17}$ , and  $G_{21}$  along the same edge.
2. There is no non-structural mass.
3. The quadrilateral faces need not be planar.
4. Stresses are given in the basic coordinate system.
5. All four continuations are required.
6. No physical property of this element can be used as a local design variable in automated design.

**Input Data Entry**    **CMASS1**    Scalar Mass Connection

**Description:** Defines a scalar mass element of the structural model.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
<u>CMASS1</u>	<u>EID</u>	<u>PID</u>	<u>G1</u>	<u>C1</u>	<u>G2</u>	<u>C2</u>	<u>TMAX</u>		
CMASS1	32	6	2	1					

**Field**

**Contents**

EID	Element identification number (Integer > 0).
PID	Identification number of a PMASS property entry (Default is EID) (Integer > 0).
G1,G2	Geometric grid point identification number (Integer > 0).
C1,C2	Component number (6 > Integer > 0).
TMAX	The maximum mass value allowed in design.

**Remarks:**

1. Scalar points may be used for G1 and/or G2, in which case the corresponding C1 and/or C2 must be zero or blank. Zero or blank may be used to indicate a grounded terminal G1 or G2 with a corresponding blank or zero C1 or C2. A grounded terminal is a point whose displacement is constrained to zero.
2. The two connection points (G1, C1) and (G2, C2), must be distinct. Except in unusual circumstances, one of them will be a grounded terminal with blank entries for G and C.
3. The TMAX value is used only for shape function design variable linking.

Input Data Entry    CMASS2    Scalar Mass Property and Connection

Description:    Defines a scalar mass element of the structural model without reference to a property entry.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CMASS2	EID	M	G1	C1	G2	C2	TMIN	TMAX	
CMASS2	2	9.25	6	1					

Field

Contents

EID                    Element identification number (Integer > 0).

M                     The value of the scalar mass (Real).

G1,G2                Geometric grid point identification number (Integer > 0).

C1,C2                Component number 6 > Integer > 0).

TMIN,TMAX           The minimum and maximum mass values in design.

Remarks:

1. Scalar points may be used for G1 and/or G2, in which case the corresponding C1 and/or C2 must be zero or blank. Zero or blank may be used to indicate a grounded terminal G1 or G2 with a corresponding blank or zero C1 or C2. A grounded terminal is a point whose displacement is constrained to zero.
2. This single card completely defines the element since no material or geometric properties are required.
3. The two connection points (G1, C1) and (G2, C2), must be distinct. Except in unusual circumstances, one of them will be a grounded terminal with blank entries for G and C.
4. The TMIN and TMAX values are used only for shape function design variable linking.



**Input Data Entry**    CONM1    Concentrated Mass Element Connection, General Form

**Description:**    Defines a 6 x 6 symmetric matrix at a geometric grid point of the structural model.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CONM1	EID	G	CID	M11	M21	M22	M31	M32	CONT
CONM1	2	22	2	2.9		6.3			+1
CONT	M33	M41	M42	M43	M44	M51	M52	M53	CONT
+1	4.8				28.6				+2
CONT	M54	M55	M61	M62	M63	M64	M65	M66	
+2		28.6						28.6	

**Field**

**Contents**

EID                    Element identification number (Integer > 0).

G                      Grid point identification number (Integer > 0).

CID                    Coordinate system identification number for the mass matrix (Integer > 0).

Mij                    Mass matrix values (Real).

**Remarks**

1. For a less general means of defining concentrated mass at grid points, see CONM2.
2. No physical property of this element can be used as a local design variable for automated design.

Input Data Entry    CONM2    Concentrated Mass Element Connection, Rigid Body Form

Description:    Defines a concentrated mass at a grid point of the structural model.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CONM2	EID	G	CID	M	X1	X2	X3		CONT
CONM2	2	15	6	49.7					123

CONT	I11	I21	I22	I31	I32	I33	TMIN	TMAX	
+23	16.2		16.2			7.8			

<u>Field</u>	<u>Contents</u>
EID	Element identification number (Integer > 0).
G	Grid point identification number (Integer > 0).
CID	Coordinate system identification number (Integer > 0). A CID of -1 (integer) allows the user to input X1, X2, X3 as the center of gravity location in the <u>basic</u> coordinate system.
M	Mass value (Real).
X1,X2,X3	Offset distances from the grid point to the center of gravity of the mass in the coordinate system defined in field 4, unless CID = -1, in which case X1, X2, X3 are the coordinates of the center of gravity of the mass in the basic coordinate system (Real).
Iij	Mass moments of inertia measured at the mass c.g., in coordinate system defined by field 4 (Real). If CID = -1, the basic coordinate system is implied.
TMIN,TMAX	The minimum and maximum mass values for design.

Remarks

1. The continuation entry may be omitted.
2. If CID = -1, offsets are internally computed as the difference between the grid point location and X1, X2, X3. The grid point locations may be defined in a nonbasic coordinate system. In this case, the values of Iij must be in a coordinate system that parallels the basic coordinate system.

3. The form of the inertia matrix about its c.g. is taken as:

$$\begin{bmatrix} M & & & & \\ & M & & & \\ & & M & & \\ & & & I_{11} & \\ & & & -I_{21} & I_{22} \\ & & & -I_{31} & -I_{32} & I_{33} \end{bmatrix} \quad \text{SYM}$$

where  $M = \int \rho dv$

$$I_{11} = \int \rho (x_2^2 + x_3^2) dv$$

$$I_{22} = \int \rho (x_1^2 + x_3^2) dv$$

$$I_{33} = \int \rho (x_1^2 + x_2^2) dv$$

$$I_{21} = \int \rho x_1 x_2 dv$$

$$I_{31} = \int \rho x_1 x_3 dv$$

$$I_{32} = \int \rho x_2 x_3 dv$$

and  $x_1, x_2, x_3$  are components of distance from the c.g. in the coordinate system defined in Field 4. The negative signs for the off-diagonal terms are supplied by the program. A warning message is issued if the inertia matrix is non-positive definite, as this may cause fatal errors in dynamic analysis modules.

4. For design, the mass moments of inertia must be zero.
5. The TMIN and TMAX values are used only for shape function design variable linking.

Input Data Entry    CONROD    Rod Element Property and Connection

Description:    Defines a rod element of the structural model without reference to a property entry.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CONROD	EID	G1	G2	MID	A		C	NSM	CONT
CONROD	2	16	17	23	2.69				
CONT	TMIN	TMAX							

Field

Contents

EID                    Element identification number (Integer > 0).

G1,G2                Grid point identification numbers of connection points (Integer > 0)

MID                   Material identification number (Integer > 0).

A                     Area of rod (Real).

J                      Torsional constant (Real).

C                      Coefficient for torsional stress determination (Real).

NSM                   Nonstructural mass per unit length (Real).

TMIN,TMAX           Minimum and maximum allowable cross-sectional areas in design (Real > 0.0 or blank)

Remarks:

1. For structural problems, CONROD entries may only reference MAT1 material entries.
2. The continuation entry is optional.
3. TMAX and TMIN are ignored unless element is linked to global design variable through an ELIST entry.

Input Data Entry CONVERT

Description: Defines conversion factors for various physical quantities.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CONVERT	QUANT1	FACTOR	QUANT2	FACTOR	QUANT3	FACTOR	QUANT4	FACTOR	CONT
CONVERT	MASS	0.00259							
CONT	QUANT	FACTOR	QUANT	FACTOR	-etc-				

Field

Contents

QUANT1            A character string identifying the physical quantity to be converted

                 - MASS, VELOCITY

FACTOR           The conversion factor (Real ] 0.0)

Remarks:

1. Any number of valid quantity-factor combinations can be entered on a single entry.
2. Only MASS and VELOCITY are currently valid quantity entries.
3. Input mass values will be multiplied by the input factor.  
Input velocities will be multiplied by the factor.

Input Data Entry    CORD1C    Cylindrical Coordinate System Definition,  
Form 1.

Description:    Defines a cylindrical coordinate system by reference to three grid points.    These points must be defined in coordinate systems whose definition does not involve the coordinate system being defined.    The first point is the origin, the second lies on the z-axis, and the third lies in the plane of the azimuthal origin.

Format and Examples:

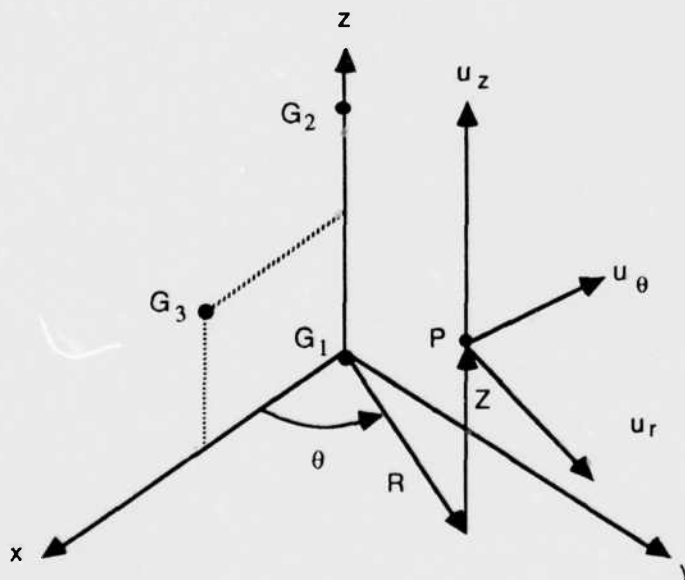
1	2	3	4	5	6	7	8	9	10
CORD1C	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1C	3	16	32	19					

Field

Contents

CID                      Coordinate system identification number (Integer > 0)

G1, G2, G3              Grid point identification number (Integer > 0; G1 ≠ G2 ≠ G3).



Remarks:

1. Coordinate system identification numbers on all CORD1R, CORD1C, CORD1S, CORD2R, CORD2C, and CORD2S entries must be unique.
2. The three points G1, G2, and G3 must be noncollinear.
3. The location of a grid point (P in the sketch) in this coordinate system is given by (R,  $\theta$ , Z) where  $\theta$  is measured in degrees.

4. The displacement coordinate directions at P are dependent on the location of P as shown above by  $(u_r, u_\theta, u_z)$ .
5. Points on the z-axis may not have their displacement directions defined in this coordinate system since an ambiguity results.
6. One or two coordinate systems may be defined on a single entry.

Input Data Entry CORD1R Rectangular Coordinate System Definition, Form 1.

**Description:** Defines a rectangular coordinate system by reference to three grid points. These points must be defined in coordinate systems whose definition does not involve the coordinate systems defined. The first point is the origin, the second lies on the z-axis, and the third lies in the x-z plane.

**Format and Examples:**

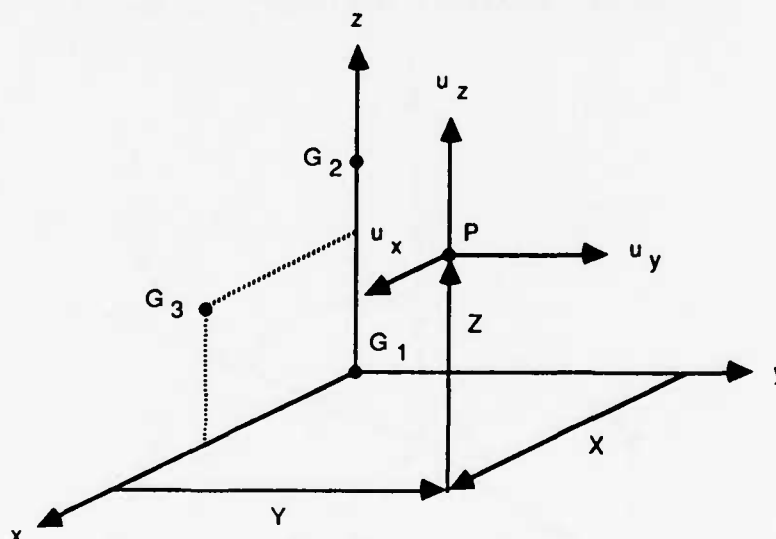
1	2	3	4	5	6	7	8	9	10
CORD1R	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1R	3	16	32	19					

**Field**

**Contents**

CID                      Coordinate system identification number (Integer > 0)

G1, G2, G3              Grid point identification number (Integer > 0; G1 ≠ G2 ≠ G3).



**Remarks:**

1. Coordinate system identification numbers on all CORD1R, CORD1C, CORD1S, CORD2R, CORD2C, and CORD2S entries must be unique.
2. The three points G1, G2, and G3 must be noncollinear.
3. The location of a grid point (P in the sketch) in this coordinate system is given by (X, Y, Z).
4. The displacement coordinate directions at P are shown above by (ux, uy, uz).
5. One or two coordinate systems may be defined on a single entry.



Input Data Entry CORD1S Spherical Coordinate System Definition, Form 1.

**Description:** Defines a spherical coordinate system by reference to three grid points. These points must be defined in coordinate systems whose definition does not involve the coordinate systems defined. The first point is the origin, the second lies on the z-axis, and the third lies in the plane of the azimuthal origin.

**Format and Examples:**

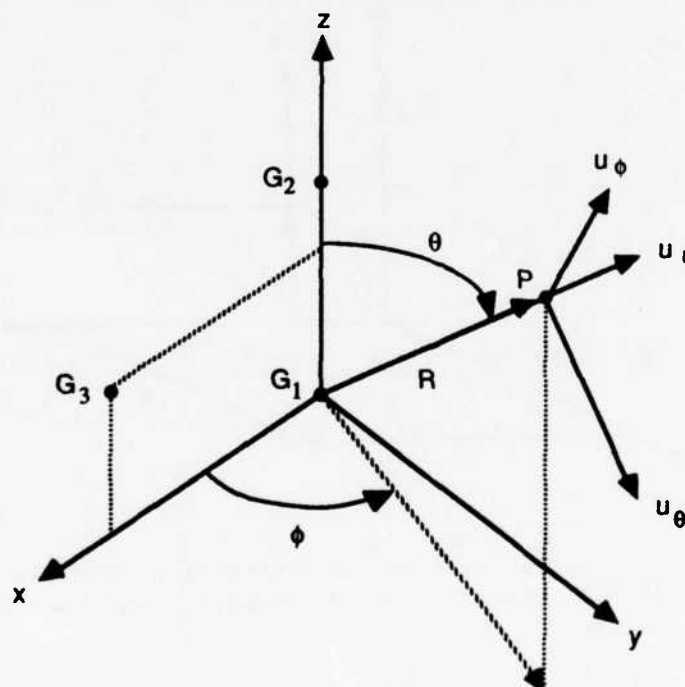
1	2	3	4	5	6	7	8	9	10
CORD1S	CID	G1	G2	G3	CID	G1	G2	G3	
CORD1S	3	16	32	19					

**Field**

**Contents**

CID                      Coordinate system identification number (Integer > 0)

G1, G2, G3              Grid point identification number (Integer > 0; G1 ≠ G2 ≠ G3).



**Remarks:**

1. Coordinate system identification numbers on all CORD1R, CORD1C, CORD1S, CORD2R, CORD2C, and CORD2S entries must be unique.
2. The three points G1, G2, and G3 must be noncollinear.
3. The location of a grid point (P in the sketch) in this coordinate system is given by (R, θ, φ) where θ and φ are measured in degrees.

4. The displacement coordinate directions at P are dependent on the locations of P as shown above by  $(u_r, u_\theta, u_\phi)$ .
5. Points on the polar axis may not have their displacement direction defined in this coordinate system since an ambiguity results.
6. One or two coordinate systems may be defined on a single entry.

Input Data Entry CORD2C Cylindrical Coordinate System Definition, Form 2.

**Description:** Defines a cylindrical coordinate system by reference to the coordinates of three grid points. The first point defines the origin. The second point defines the direction of the z-axis. The third lies in the plane of the azimuthal origin. The reference coordinate system must be independently defined.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CORD2C	CID	RID	A1	A2	A3	B1	B2	B3	CONT
CORD2C	3	17	-2.9	1.0	0.0	3.6	0.0	1.0	123
CONT	C1	C2	C3						
+23	5.2	1.0	-2.9						

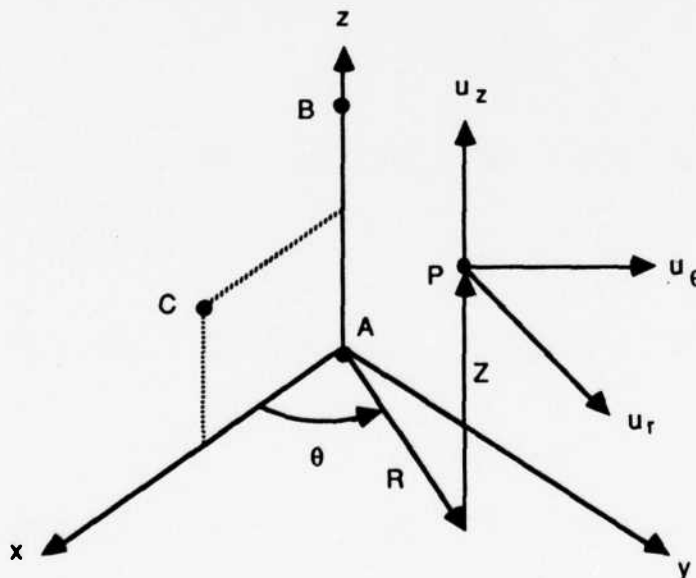
**Field**

**Contents**

CID Coordinate system identification number (Integer > 0)

RID Reference to a coordinate system which is defined independently of new coordinate system (Integer  $\geq 0$  or blank)

A1,A2,A3  
B1,B2,B3  
C1,C2,C3 Coordinates of three points in coordinate system defined in field (Real).



**Remarks:**

1. Continuation card must be present.
2. The three points (A1, A2, A3), (B1,B2,B3), (C1,C2,C3) must be unique and noncollinear.

3. Coordinate system identification numbers on CORD1R, CORD1C, CORD1S, CORD2R, CORD2C, and CORD2S cards must all be unique.
4. An RID of zero references the basic coordinate system.
5. The location of a grid point (P in the sketch) in this coordinate is given by  $(R, \theta, Z)$  where  $\theta$  is measured in degrees.
6. The displacement coordinate directions at P are dependent on the location of P as shown above by  $(u_r, u_\theta, u_z)$ .
7. Points on the z-axis may not have their displacement direction defined in this coordinate system since an ambiguity results.

# Input Data Entry CORD2R

## Rectangular Coordinate System Definition, Form 2.

**Description:** Defines a rectangular coordinate system by reference to coordinates of three points. The first point defines the origin. The second defines the direction of the z-axis. The third point defines a vector which, with the z-axis, defines the x-z plane. The reference coordinate system must be independently defined.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
CORD2R	CID	RID	A1	A2	A3	B1	B2	B3	CONT
CORD2R	3	17	-2.9	1.0	0.0	3.6	0.0	1.0	123
CONT	C1	C2	C3						
+23	5.2	1.0	-2.9						

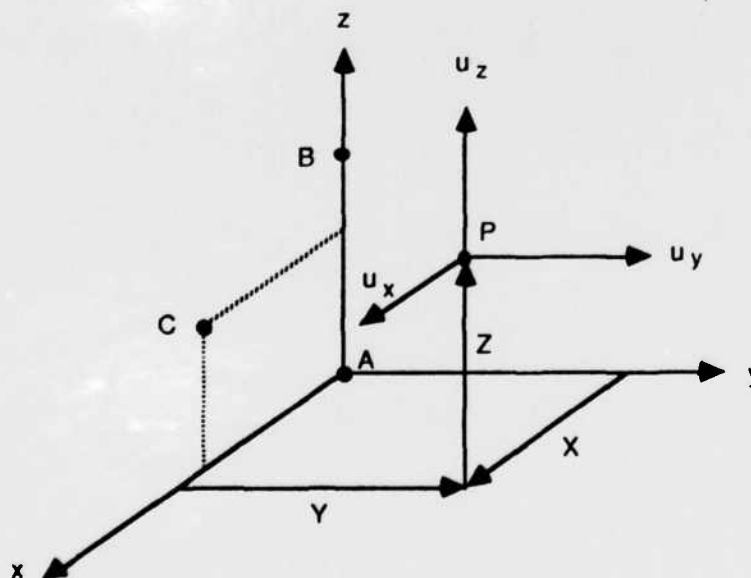
### Field

### Contents

CID                      Coordinate system identification number (Integer > 0)

RID                      Reference to a coordinate system which is defined independently of new coordinate system (Integer ≥ 0 or blank)

A1,A2,A3  
B1,B2,B3  
C1,C2,C3                      Coordinates of three points in coordinate system defined in field 3 (Real)



Remarks:

1. The continuation card must be present.
2. The three points (A1, A2, A3), (B1, B2, B3), (C1, C2, C3) must be unique and noncollinear.
3. Coordinate system identification numbers on all CORD1R, CORD1C, CORD1S, CORD2R, CORD2C, and CORD2S cards must be unique.
4. An RID of zero references the basic coordinate system.
5. The location of a grid point (P in the sketch) in this coordinate system is given by (X, Y, Z).
6. The displacement coordinate directions at P are shown by ( $u_x$ ,  $u_y$ ,  $u_z$ ).

Input Data Entry CORD2S Spherical Coordinate System Definition, Form 2.

**Description:** Defines a spherical coordinate system by reference to the coordinates of three points. The first point defines the origin. The second point defines the direction of the z-axis. The third lies in the plane of the azimuthal origin. The reference coordinate system must be independently defined.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CORD2S	CID	RID	A1	A2	A3	B1	B2	B3	CONT
CORD2S	3	17	-2.9	1.0	0.0	3.6	0.0	1.0	123
CONT	C1	C2	C3						
+23	5.2	1.0	-2.9						

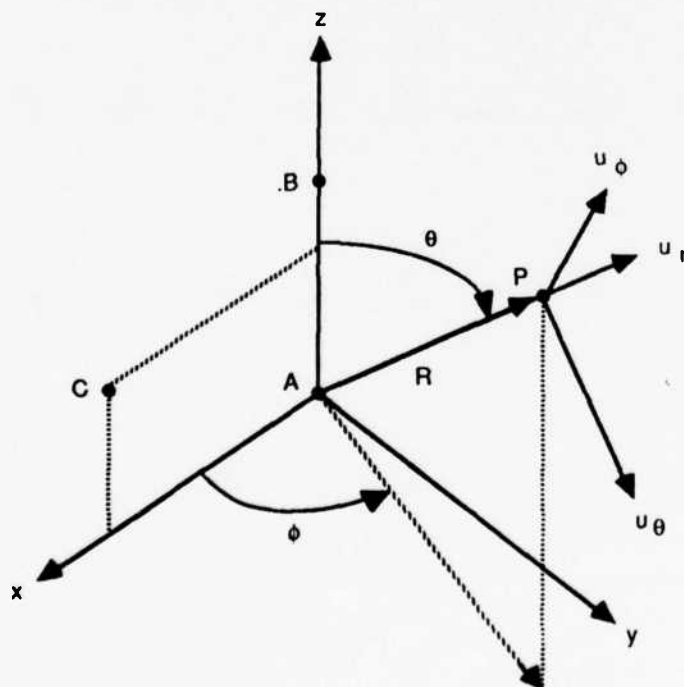
**Field**

**Contents**

CID Coordinate system identification number (Integer > 0)

RID Reference to a coordinate system which is defined independently of of new coordinate system (Integer  $\geq 0$  or blank)

A1,A2,A3  
B1,B2,B3  
C1,C2,C3 Coordinates of three points in coordinate system defined in field 3 (Real)



Remarks:

1. The continuation card must be present.
2. The three points (A1, A2, A3), (B1, B2, B3), (C1, C2, C3) must be unique and noncollinear.
3. Coordinate system identification numbers on all CORD1R, CORD1C, CORD1S, CORD2R, CORD2C and CORD2S cards must be unique.
4. An RID of zero references the basic coordinate system.
5. The location of a grid point (P in the sketch) in this coordinate system is given by (R,  $\theta$ ,  $\phi$ ) where  $\theta$  and  $\phi$  are measured in degrees.
6. The displacement coordinate directions at P are shown above by ( $u_r$ ,  $u_\theta$ ,  $u_\phi$ ).
7. Points on the polar axis may not have their displacement directions defined in this coordinate system since an ambiguity results.



**Input Data Entry**    CODMEM1    Isoparametric Quadrilateral Element Connection

**Description:** Defines the isoparametric quadrilateral membrane element.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
CODMEM1	EID	PID	G1	G2	G3	G4	TH	TMAX	
CODMEM1	72	13	13	14	15	16	29.2		

**Field**

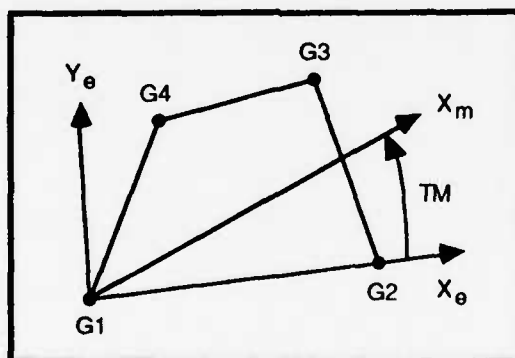
**Contents**

**EID**                      Element identification number (Integer > 0).

**PID**                      Identification number of a PQDMEM1 property card (Default is EID) (Integer > 0).

**G1,G2,G3,G4**            Grid point identification numbers of connection points (Integer > 0)

**TH**                        Material property orientation angle. If TH is real, the sketch below gives the sign convention for TH. If TH is an integer, the material x-axis is along the x-axis of coordinate system identified by the integer.



**TMAX**                      Maximum allowable element thickness in design (Real > 0.0 or blank).

**Remarks:**

1. Grid points G1 through G4 must be ordered consecutively around the perimeter of the element.
2. All interior angles must be less than 180°.
3. TMAX is ignored unless element is linked to global design variable by an ELIST entry.

# Input Data Entry    COUAD4    Quadrilateral Element Connection

Description:    Quadrilateral plate element (QUAD4) of the structural model.  
This is an isoparametric membrane-bending element.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
COUAD4	EID	PID	G1	G2	G3	G4	TM	ZOFF	CONT
COUAD4	101	17	1001	1005	1010	1024	45.0	0.01	ABC
CONT		TMAX	T1	T2	T3	T4			
+BC			0.03	0.125	0.05	0.04			

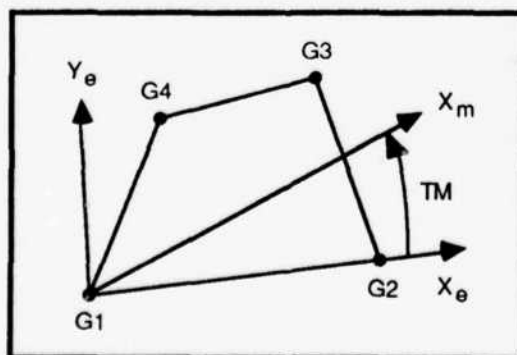
## Field

## Contents

EID	Element identification number (Integer > 0)
PID	Identification number of a PSHELL, PCOMP, PCOMP1 or PCOMP2 entry (Default is EID) (Integer > 0).
Gi	Grid point identification numbers of connection points (Integer > 0).
ZOFF	Offset of the element mid plane from the plane of grid points (Real or blank, see Remark 2 for default).
TM	Material property orientation specification (Real or blank; or $0 \leq \text{Integer} < 1,000,000$ ). If Real or blank, specifies the material property orientation angle in degrees. If Integer, the orientation of the material x-axis is along the projection onto the plane of the element of the x-axis of the coordinate system specified by the integer value.
TMAX	Maximum allowable element thickness in design (Real > 0.0).
Ti	Membrane thickness of element at grid points Gi (Real or blank, see Remark 3 for default).

## Remarks:

1. The QUAD4 geometry, coordinate systems and numbering are showed in the figure below:



2. The material coordinate system (TM) and the offset (ZO) may also be provided on the PSHELL entry for non-composite elements. The property data will be used if the corresponding field on the CQUAD4 entry is blank.
3. The Ti are optional, if not supplied they will be set to the value of T specified on the PSHELL entry. In such cases, the continuation entry is not required.
4. TMAX is ignored unless the element is linked to the global design variables by an ELIST entry.

Input Data Entry CROD Rod Element Connection

Description: Defines a tension-compression-torsion element (ROD) of the structural model.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CROD	EID	PID	G1	G2	TMAX				
CROD	12	13	21	23					

Field

Contents

EID	Element identification number (Integer > 0).
PID	Identification number of a PROD property entry (Default is EID) (Integer > 0).
G1,G2	Grid point identification numbers of connection points (Integer > 0)
TMAX	Maximum allowable rod area in design (Real > 0.0 or Blank).

Remarks:

1. See CONROD for alternative method of rod definition.
2. Only one ROD element may be defined on a single entry.
3. TMAX is ignored unless the element is linked to global design variables by an ELIST entry.

Input Data Entry CSHEAR Shear Panel Element Connection

Description: Defines a shear panel element (SHEAR) of the structural model.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CSHEAR	EID	PID	G1	G2	G3	G4	TMAX		
CSHEAR	3	6	1	5	3	7			

Field

Contents

EID                    Element identification number (Integer > 0).

PID                    Identification number of a PSHEAR property entry (Default is EID) (Integer > 0).

G1,G2,G3,G4          Grid point identification numbers of connection points (Integer > 0)

TMAX                   Maximum allowable thickness in design (Real > 0.0 or blank).

Remarks:

1. Grid points G1 through G4 must be ordered consecutively around the perimeter of the element.
2. All interior angles must be less than 180°.
3. TMAX is ignored unless element is linked to global design variables by an ELIST entry.

Input Data Entry CTRMEM

Description: Defines TRMEM constant strain triangular membrane element.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
CTRMEM	EID	PID	G1	G2	G3	THETA	TMAX		
CTRMEM	100	500	1	7	12				

Field

Contents

EID	Element identification number (Integer > 0).
PID	Identification of PTRMEM entry (Integer > 0) Default - EID.
G1,G2,G3	Grid point identifications of connection points (Integer > 0).
THETA	Material orientation angle (Real) <u>OR</u> 0 < Integer < 1,000,000. If integer, then material x-axis lies along x-axis of coordinate system identified by the integer.
TMAX	Maximum allowable thickness in design.

Remarks:

1. The TMAX value is used only for shape function design variable linking.

# Input Data Entry DCONALE

**Description:** Defines an aileron effectiveness constraint, of the form:

$$AE \leq AEREQ \text{ (UPPER) or } AE \geq AEREQ \text{ (LOWER)}$$

where,

$$AE = \frac{-C_{\delta a}}{C_{\frac{Pb}{2v}}}$$

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
DCONALE	SID	CTYPE	AEREQ						
DCONALE	25	LOWER	0.4						

## Field

## Contents

SID	Aerodynamic set identification for the imposed constraint (Integer > 0).
CTYPE	Constraint type; either UPPER for upper bound or LOWER for lower bound (Text, Def = LOWER).
AEREQ	Required aileron effectiveness. (Real ≠ 0.0)

## Remarks:

1. The aileron effectiveness constraint will only be applied if called out in the Solution Control and if a lateral, zero degree of freedom aeroelastic trim analysis is being performed.
2. A LOWER bound constraint excludes all values to the left of AEREQ on a real number line, while an UPPER bound constraint excludes all values to the right, irrespective of the sign of AEREQ.

Input Data Entry DCONCLA

Description: Defines a flexible lift curve slope constraint of the form:

$$CLA \leq CLAREQ \text{ or } CLA \geq CLAREQ$$

where,

$$CLA = (C_{L\alpha})_f / (C_{L\alpha})_r$$

the ratio of flexible to rigid lift curve slopes.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
DCONCLA	SID	CTYPE	CLAREQ						
DCONCLA	25	UPPER	0.8						

Field

Contents

SID	Aerodynamic set identification for the imposed constraint (Integer > 0)								
CTYPE	Constraint type; either UPPER for upper bound or LOWER for lower bound (Text, Def = LOWER).								
CLAREQ	Required flexible-to-rigid lift curve slope. (Real ≠ 0.0)								

Remarks:

1. This entry imposes a lift effectiveness only if called out in the Solution Control and if a lift or lift/pitching moment aeroelastic trim analysis is being performed.
2. A LOWER bound constraint excludes all values to the left of CLAREQ on a real number line, while an UPPER bound constraint excludes all values to the right, irrespective of the sign of CLAREQ.



# Input Data Entry DCONDSP

**Description:** Defines a deflection constraint of the form:

$$A_{juj} \leq \delta_{all} \text{ (UPPER BOUND) or } A_{juj} \geq \delta_{all} \text{ (LOWER BOUND)}$$

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
DCONDSP	CTSET	DCID	CTYPE	DALL	LABEL	G	C	A	CONT
DCONDSP	1	10	LOWER	-2.3	TIP	32	3	2.0	ABC
CONT		G	C	A	G	C	A		etc
+BC		7	3	-4.0					

## Field

## Contents

CTSET	Constraint set identification number (Integer).
DCID	Constraint identification number (Integer).
CTYPE	Constraint type, either UPPER or LOWER bound (Text, Def - UPPER).
DALL	Allowable displacement (Real).
LABEL	User specified label to identify constraint.
G	Grid identification.
C	Component number - any one of digits 1-6.
A	Real coefficient.

## Remarks:

- Both upper and lower bounds on the deflections can be specified by this entry. E.g., if constraints of the form  $|u| \leq 2.0$  are to be imposed, one DCONDSP entry would use CTYPE = UPPER, DALL = 2.0, G = 32, C = 3, A = 1.0 while a second entry would use CTYPE = LOWER, DALL = -2.0, G = 32, C = 3, A = 1.0.
- Twist constraints can be specified by differencing two displacements while camber constraints can be expressed as a weighted sum of three displacements.
- Any number of continuation cards are permitted.
- A LOWER bound constraint excludes all values to the left of DALL on a real number line, while an UPPER bound constraint excludes all values to the right, irrespective of the sign of DALL.

# Input Data Entry DCONFLT

Description: Defines a flutter constraint in the form of a table:

$$(\gamma - \gamma_{REQ}) / (GFACT) \leq 0.0$$

Format and Examples:

1	2	3	4	5	6	7	8	9	10
DCONFLT	SID	GFACT	V1	GAM1	V2	GAM2	V3	GAM3	CONT
DCONFLT	2		100.0	-.01	1000.0	0.0	1500.0	0.0	+ABC
CONT	V4	GAM4	V5	-etc-					
+BC									

## Field

## Contents

SID                      Constraint set identification, the constraints are referenced by the design constraint id in solution control.

GFACT                    Constraint scaling factor (Real > 0.0, D = 0.10).

V<sub>i</sub>                        Velocity value (Real).

GAM<sub>i</sub>                    Required damping value (Real).

## Remarks:

1. A negative value of GAM<sub>i</sub> refers to a stable system.
2. The V<sub>i</sub> must be in either ascending or descending order.
3. Linear interpolation is used to determine GAMA for a given velocity.
4. At least two pairs must be entered.
5. Jumps between two points (V<sub>i</sub> - V<sub>i+1</sub>) are allowed, but not at the end points.

**Input Data Entry** DCONFRO

**Description:** Defines a frequency constraint of the form:

$$f \leq f_{all} \text{ or } f \geq f_{all}$$

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
DCONFRO	SID	MODE	CTYPE	FRQALL					
DCONFRO	3	1	LOWER	6.0					

**Field**

**Contents**

**SID** Constraint set identification (Integer).

**MODE** Modal number of the frequency to be constrained (Integer).

**CTYPE** Constraint type, either UPPER for upper bounds or LOWER for lower bounds (Text, Def = LOWER).

**FRQALL** Frequency constraint (in Hz.). (Real > 0.0)

**Remarks:**

1. More than one constraint can be placed on a mode.

Input Data Entry DCONSTR

Description: Defines stress/strain constraints.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
DCONSTR	MID	CRIT	MID	CRIT	MID	CRIT	MID	CRIT	
DCONSTR	1	VMISES	10	VMISES					

Field

Contents

MID Material identification number for the constrained elements.

CRIT Failure criterion to be used (Text)

Remarks:

1. Allowable constraint criteria (CRIT) are: VMISES, TSAIWU, STRAIN
  - (A) von Mises stress constraint. Yield values are given by ST, SC and SS values on a MAT1 or MAT2 data entry.
  - (B) Tsai-Wu stress constraint. Yield values are given on the MAT8 data entry.
  - (C) Maximum strain constraint. Strain allowables for tension, compression and shear are given defined in the ST, SC and SS fields of a MAT1, MAT2 or MAT8 data entry. The shear strain allowable is used only for the shear element and is ignored for other element types.

Input Data Entry DCONTHK Thickness constraints

**Description:** Defines a list of elements (linked using ELIST entries) for which thickness constraints are to be retained on all design iterations.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
DCONTHK	ETYPE	EID	EID	EID	EID	EID	EID	EID	CONT
DCONTHK	ODMEM1	100	101	200	205				
CONT	EID	EID	-etc-						

**Alternate Form:**

1	2	3	4	5	6	7	8	9	10
DCONTHK	ETYPE	EID	"THRU"	EID					
DCONTHK	ODMEM1	100	"THRU"	200					

**Field**

**Contents**

**ETYPE** Character input identifying the element type. One of the following:

BAR  
CONM2  
ELAS  
MASS  
QDMEM1  
QUAD4  
ROD  
SHEAR  
TREMEM

**EID** Element identification number (Integer > 0 or blank)

**Remarks:**

1. The purpose of this bulk data list is to ensure that adequate physical move limits are retained in optimization with shape function design variable linking without requiring retention of all move limits. For problems with large numbers of local variables using shape functions, the move limits often cause too many minimum thickness constraints (see Remark 2) to be retained in the optimization task. Using this bulk data entry to name "critical" minimum gauge constraints (see Remark 3) will cause only the named elements' thickness constraints to be computed and retained.

NOTE that an element with a violated minimum gauge constraint will always be computed irrespective of the DCONTHK entries, but may be deleted in the constraint deletion.

2. The global design variable in shape function linking is non-physical and no reasonable restriction for a global move limit (side constraint) can be defined; therefore, constraints on the local design variables controlled by shape functions are generated by ASTROS to ensure that the design is reasonable (ie, non-negative thicknesses).
3. The DCONTHK entry should select a minimum number of elements linked to shape functions that will enable the optimizer to select physically reasonable designs without retaining all the minimum thickness constraints (potentially a very large number). Typically, this means  $N+1$  elements spread over the range of the shape function (e.g. span or chord) where  $N$  is the order of the shape ( $N=0$ , UNIFORM:  $N=1$ , LINEAR, etc.)

## Input Data Entry DESELM

Description: Designates design variable properties when the design variable is uniquely associated with a single finite element.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
DESELM	DVID	EID	ETYPE	VMIN	VMAX	VINIT	LAYERNUM	LABEL	
DESELM	5	10	CROD	0.01	10.0	1.0			

<u>Field</u>	<u>Contents</u>
DVID	Design variable identification (Integer > 0).
EID	Element identification (Integer > 0).
ETYPE	Element type.
VMIN	Minimum allowable value of the design variable (Real > 0) (Default = .001).
VMAX	Maximum allowable value of the design variable (Real > 0) (Default = 1000.)
VINIT	Initial value of the design variable (Real, $VMIN \leq VINIT \leq VMAX$ ).
LAYERNUM	The layer number if a composite element is to be designed.
LABEL	Optional user-supplied label to define the design variable (Text)

### Remarks:

1. Valid ETYPE's are CROD, CONROD, CBAR, CSHEAR, CTRMEM, CQDMEM1, CQUAD4, CMASS1, CMASS2 and CONM2.
2. The initial element size used in the structural analysis is the product of the VINIT value and the element size on the associated property entry.

# Input Data Entry DESVAR

Description: Designates design variable properties.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
DESVAR	DVID	VMIN	VMAX	VINIT	LAYERNUM	LABEL			
DESVAR	1	0.01	2.0	1.0	13	INBDTOP			

## Field

## Contents

DVID	Design variable identification (Integer > 0).
VMIN	Minimum allowable value of the design variable (Real > 0) (Default = 0.001).
VMAX	Maximum allowable value of the design variable (Real > 0) (Default = 1000.0).
VINIT	Initial value of the design variable (Real, $VMIN \leq VINIT \leq VMAX$ ).
LAYRNUM	Layer number if referencing composite element(s).
LABEL	Optional user supplied label to define the design variable (Text).

## Remarks:

1. The elements linked to the DESVAR are specified using either a PLIST or an ELIST data entry.
2. Shape function linking (using ELIST entries) will override VMIN and VMAX with large negative and positive values, respectively.



## Input Data Entry DLAGS

**Description:** This entry is used in conjunction with RLOAD1, RLOAD2, TLOAD1 and TLOAD2 data entries and defines time lags and phase lags as well as the set identification of the static load.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
DLAGS	SID	LID	TAU	PHASE	LID	TAU	PHASE		
DLAGS	5	21	0.04	20.0	10	0.0	45.0		

### Field

### Contents

SID	Identification number of DLAGS set (Integer > 0)
LID	Identification number of time (or frequency) independent applied load (Integer > 0)
TAU	Time delay for the designated load set (Real)
PHASE	Phase lag (in degrees) for the designated load set (Real)

### Remarks:

1. One or two dynamic load sets may be defined on a single entry.
2. Refer to RLOAD1, RLOAD2, TLOAD1 or TLOAD2 entries for formulas which define the manner in which TAU and PHASE are used.
3. The phase parameter is used only in conjunction with RLOAD1 and RLOAD2 data entries.
4. The LID set can refer to statically applied loads as well as to additional dynamic loads input on DLONLY entries.
5. TAU and PHASE can be defaulted to zero, but LID must not be zero.

# Input Data Entry     DLOAD

**Description:** Defines a dynamic loading condition for frequency response or transient response problems as a linear combination of load sets defined via RLOAD1 or RLOAD2 entries (for frequency response) or TLOAD1 or TLOAD2 entries (for transient response).

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
DLOAD	SID	S	S1	L1	S2	L2	S3	L3	CONT
DLOAD	17	1.0	2.0	6	-2.0	7	2.0	8	+A
CONT	S4	L4		etc					
+A	-2.0	9							

## Field

## Contents

SID                    Load set identification number (Integer > 0)

S                      Scale factor (Real)

Si                     Scale Factors (Real)

Li                     Load set identification numbers defined via bulk data entries enumerated above (Integer > 0)

## Remarks:

1. The load vector being defined by this entry is given by
$$[P] = S \sum_i S_i L_i$$
2. The Li must be unique.
3. SID must be unique from all Li.
4. TLOAD1 and TLOAD2 loads may be combined only through the use of the DLOAD entry.
5. RLOAD1 and RLOAD2 loads may be combined only through the use of the DLOAD entry.
6. SID must be unique for all TLOAD1, TLOAD2, RLOAD1 and RLOAD2 entries.
7. Load sets must be selected by a solution control command (DLOAD = SID).

Input Data Entry DLONLY

Description: This entry is used in conjunction with the RLOAD1, RLOAD2, TLOAD1 and TLOAD2 entries and defines the point where the dynamic load is to be applied with the scale factor A.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
DLONLY	SID	P	C	A	P	C	A		
DLONLY	3	6	2	8.2	15	1	10.1		

Field

Contents

SID Identification number of DLONLY set (Integer > 0)

P Grid, extra point or scalar point identification number (Integer > 0)

C Component number (1-6 for grid points; blank or 0 for extra points or scalar points).

A Load factor A for the designated coordinate (Real)

Remarks:

1. One or two load factors may be defined on a single entry.
2. Refer to RLOAD1, RLOAD2, TLOAD1 or TLOAD2 entries for the formulas which define the load factor A.
3. Component numbers refer to global coordinates.

# Input Data Entry DMI Direct Matrix Input

**Description:** Used to input matrix data base entities directly. Generates a matrix of the form:

$$[A] = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & \dots & \dots & A_{mn} \end{bmatrix}$$

where the elements  $A_{ij}$  may be real or complex.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
DMI	NAME	PREC	FORM	M	N				CONT
DMI	TEST	RDP	REC	3	4				ABC
CONT	C1	R1	A(R1,C1)	C2	R2	A(R1,C2)	A(R1+1,C2)	C3	CONT
+BC	1	2	2.0	2	1	3.0	4.0	4	DEF
CONT	R1	A(R1,C3)	C4	R2	A(R2,C4)				
+EF	1	5.0	4	3	6.5				

## Field

## Contents

NAME	Any valid data base entity name (Text 8)
PREC	The precision of the matrix entity to be loaded. Any one of the following character strings: RSP, CSP, RDP, CDP.
FORM	The form of the matrix entity to be loaded. Any one of the following: REC, SYM, DIAG, IDENT.
M	The number of rows in the matrix.
N	The number of columns in the matrix.
$C_i$	The column number of the column being loaded.
$R_i$	The row number of the first row in the string being loaded.
$A(R_i, C_i)$	Matrix terms.

## Remarks:

1. If the named entity exists, it will be flushed and re-loaded. If the entity does not exist, it will be created.

2. Column and row identifiers ( $C_i$ ,  $R_i$ ) must always appear together although they can appear in any two contiguous fields.
3. Columns must be loaded in increasing column number order. If more than one string is to be loaded for a particular column, the  $C_i$  field must contain the same value as in the previous string. Strings must be loaded in increasing row order without overlap.
4. Complex matrices require two real values for each matrix term. These can be split across physical entry boundaries.

# Input Data Entry DMIG Direct Matrix Input at Grid Points

**Description:** Defines structure-related direct input matrices with terms located by external grid/component values.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
DMIG	NAME	PREC	FORM						CONT
DMIG	TEST	RDP	REC						ABC
CONT	GCOL	CCOL	GROW	CROW	X <sub>ij</sub>	Y <sub>ij</sub>			CONT
+BC	1001	4	2001	2	1.25+5				DEF
CONT	GCOL	CCOL	GROW	CROW	X <sub>ij</sub>	Y <sub>ij</sub>			CONT
+EF	1001	4	3001	3	2.67+4	etc			

## Field

## Contents

NAME	Any valid data base entity name (Text)
PREC	The precision of the matrix entity to be loaded. Any one of the following character strings: RSP, RDP, CSP, CDP
FORM	The form of the matrix entity to be loaded. Any one of the following: REC, SYM, DIAG, IDENT
GCOL	Grid, scalar or extra point identification for column index
CCOL	Component number for GCOL, $0 \leq \text{CCOL} \leq 6$ if GCOL is a grid point, zero or blank for scalar or extra points.
GROW	Grid, scalar or extra point identification for row index.
CROW	Component number for GROW, $0 \leq \text{CROW} \leq 6$ if GROW is a grid point, zero or blank for scalar or extra points.
X <sub>ij</sub> , Y <sub>ij</sub>	Matrix term. X <sub>ij</sub> is real part for real or complex matrices. Y <sub>ij</sub> is the imaginary part for complex matrices and is ignored for real matrices.

## Remarks:

1. If the named entity exists, it will be flushed and reloaded. If the entity does not exist, it will be created.
2. The number of rows and columns will be either p-set size or g-set size depending on whether the named entity is requested by K2PP, M2PP, B2PP or K2GG, M2GG, B2GG.

3. Each non-null term in the matrix requires a continuation entry. The column index and row index values can appear any number of times on a logical entry but a fatal error will occur if the same term is entered more than once.
4. The matrix terms can be entered in any order.

Input Data Entry DYNRED Dynamic Reduction Data

Description: Defines dynamic reduction control data.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
DYNRED	SID	FMAX	NVEC						
DYNRED	1	50.0							

Field

Contents

SID Set identification number (Integer > 0).

FMAX Highest frequency of interest (Hertz) (Real > 0 or blank).

NVEC Number of generalized coordinates desired (Integer > 0 or blank).

Remarks:

1. Dynamic reduction data must be requested in the Solution Control packet (DYNRED = SID).
2. The user should select either an FMAX, or both the FMAX and NVEC fields. FMAX should not be greater than necessary for the specific dynamic analysis. NVEC, if specified, should be significantly less than the size of the F-set to realize any computational cost savings. NVEC will limit dynamic reduction to using NVEC flexible vectors.
3. Dynamic reduction transforms the motions of the F-set to the motions of the user defined A-set plus motions of generalized coordinates created in the process. The generalized coordinates represent overall structure displacements which are approximate normal mode shapes. The generalized coordinates are identified by SCALAR points that are automatically generated. The SCALAR point ID's begin with 1 greater than the highest user GRID, SCALAR, or EXTRA point ID.



# Input Data Entry EIGR Real Eigenvalue Extraction Data

**Description:** Defines data needed to perform real eigenvalue extraction.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
EIGR	SID	METHOD	F1	F2	NE	ND		E	CONT
EIGR	13	INV	1.9	15.6		12			ABC
CONT	NORM	G	C						
+BC	POINT	32	4						

## Field

## Contents

**SID** Set identification number (Unique integer > 0)

**METHOD** Method of eigenvalue extraction, one of the BCD values, "INV" or "GIV"

INV - Inverse power method

GIV - Given's method

	When METHOD = INV	When METHOD = GIV
F1, F2	Frequency range of interest (Real $\geq$ 0.0). Both must be input.	Frequency range of interest (Real $\geq$ 0.0; F1 < F2). If ND is not blank, eigenvectors are found whose natural frequencies lie in the range between F1 and F2.
NE	Estimate of number of roots in range (Required)	Not used.
ND	Desired number of roots. (Default is 3*NE, INV only). (Integer > 0).	Desired number of eigenvectors. (Integer > 0). If ND is blank or zero, the number of eigenvectors is determined from F1 and F2. (Default = 0)

**E** Convergence test (Real, Default = 1.0E-6)

**NORM** Method for normalizing eigenvectors, one of the BCD values, "MASS", "MAX", or "POINT"

MASS - Normalize to unit value of the generalized mass

MAX - Normalize to unit value of the largest component in the analysis set (Default)

POINT - Normalize to unit value of the component defined in fields 3 and 4 of continuation (defaults to "MAX" if point is not defined)

G            Grid or scalar point identification number (Required only if  
NORM = "POINT" (Integer > 0)

C            Component number (One of the integers 1-6) (Required only if  
NORM = "POINT" and G is a geometric grid point)

Remarks:

1. Real eigenvalue extraction data sets must be selected in Solution Control (Method = SID) to be used.
2. The units of F1 and F2 are cycles per unit time.
3. The continuation entry is not required. MAX normalization is then used.
4. If METHOD = "GIV" all eigenvalues are found.
5. If METHOD = "GIV", the mass matrix for the analysis set must be positive definite. Singularities or near-singularities of the mechanism type in the mass matrix will produce poor numerical stability for the GIV method.
6. If NORM = MAX, components that are not in the analysis set may have values larger than unity.
7. If NORM = POINT, the selected component should be in the analysis set. The program uses MAX when it is not in the analysis set.
8. The desired number of roots (ND) includes all roots previously found, such as rigid body modes determined with the use of the SUPORT entry.

# Input Data Entry ELEMLIST

Description: Defines a list of elements for which outputs are desired.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
ELEMLIST	SID	ETYPE	EID	EID	EID	EID	EID	EID	CONT
ELEMLIST	100	ODMEM1	100	101	200	205			
CONT	EID	EID	-etc-						

## Alternate Form

1	2	3	4	5	6	7	8	9	10
ELEMLIST	SID	ETYPE	EID	"THRU"	EID				
ELEMLIST	100	ODMEM1	100	THRU	200				

## Field

## Contents

SID Set identification number referenced by Solution Control.  
(Integer > 0 )

ETYPE Character input identifying the element type. One of the following:

BAR  
ELAS  
IHEX1  
IHEX2  
IHEX3  
QDMEM1  
QUAD4  
ROD  
SHEAR  
TRMEM

EID Element identification number (Integer > 0 or blank)

## Remarks:

1. In order to be used, the SID must be referenced by Solution Control.
2. Nonexistent elements may be referenced and will result in no error message.
3. Any number of continuations is allowed.

# Input Data Entry ELIST

Description: Defines element connectivity entries associated with a design variable.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
ELIST	DVID	ETYPE	EID1	PREF1	EID2	PREF2	EID3	PREF3	CONT
ELIST	10	CROD	12	12.0	22	1.0			
CONT	EID4	PREF4	EID5	PREF5	-etc-				

## Field

## Contents

DVID	Design variable identification (Integer).
ETYPE	Element type associated with this list (e.g., CROD).
EID1, EID2, EIDi	Element identification numbers.
PREF <sub>i</sub>	Linking factor for the associated EID.

## Remarks:

1. Allowable ETYPES are: CROD, CONROD, CSHEAR, CQDMEM1, CQUAD4, CTRMEM, CBAR, CMASS1 and CMASS2.
2. The design variable identification must match that of a design variable defined as a DESVAR entry.
3. The linking factors define a shape function to be used as the global design variable.
4. Designed properties (e.g., thicknesses) of elements listed on ELIST entries will be set to unity to ensure proper shape function definition.

Input Data Entry    EPOINT       Extra Point List

Description:    Defines extra points of the structural model for use in dynamics problems.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
EPOINT	SETID	ID	ID	ID	ID	ID	ID	ID	CONT
EPOINT	1000	3	18	1	4	16	2		
CONT	ID	ID	ID	-etc-					

Alternate Form

EPOINT	SETID	ID1	"THRU"	ID2					
EPOINT	1000	3	THRU	18					

Field

Contents

SETID                    Extra point set identification number (Integer > 0).

ID, ID1, ID2            Extra point identification number (Integer > 0; ID1 < ID2)

Remarks:

1. The extra point set identification is selected on the BOUNDARY entry. All extra points defined with this SETID will be used in dynamic analyses in the boundary condition.
2. All extra point identification numbers must be unique with respect to all other structural and scalar points.
3. This entry is used to define coordinates used in transfer function definitions (see TF entry) and Direct Matrix Input.
4. If the alternate form is used, extra points ID1 through ID2 are defined.

## Input Data Entry      FFT

Description:    Defines parameters for controlling the Fast Fourier Transformation (FFT) during time domain response analysis.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
FET	SID	TIME	NT	RDELTF	RF	FRIM	OTYPE	FLIM	
FET	3	20.	1024	1.0					

<u>Field</u>	<u>Contents</u>
SID	FFT set identification number (Integer > 0 )
TIME	Length of time period to be analyzed (Real > 0.0)
NT	Number of time points to be used for the FFT (Integer $\geq 2$ )
RDELTF	Ratio of incremental frequency (del F) to $1 / T$ . See remarks 4 and 6. Default = 1.0.
RF	Ratio of total frequency duration (F) to $NT / 2 * T$ . See remarks 5 and 6. Default = 1.0.
FRIM	Frequency response interpolation method. Character string "LINEAR" or "CUBIC". Default is "LINEAR."
OTYPE	Type of response to be output. Character string "TIME," "FREQ" or "BOTH." Default is "TIME."
FLIM	Frequency load interpolation method. Character string "LINEAR" or "CUBIC." Default is "LINEAR."

### Remarks:

1. SID must be selected by a FFT option on a TRANSIENT command in solution control.
2. TIME is the period for periodic dynamic loads defined in the time domain. For non-periodic loads, TIME is the total time duration of the excitation plus any quiet portion desired for response decay. TIME may be larger than the time duration defined by TLOAD1 or TLOAD2 data, in which case the forcing function will be automatically set to zero for the additional time.
3. NT should be a power of 2; i.e.,  $NT = 2^m$ ,  $m = 1, 2, \dots$ ; or  $NT = 2, 4, 8, \dots$ . If NT is not a power of 2, it will be automatically set to the next highest power of 2 value.

4. The incremental frequency,  $\Delta F$ , required by the FFT algorithm, is  $1/\text{TIME}$ . The value of  $\Delta F$  may be adjusted by the user with the RDELTF factor. However, the most accurate results are normally obtained with the default case of  $\text{RDELTF} = 1.0$ .
5. The frequency duration required by the FFT algorithm is  $F = \text{NT} / 2 * T$ . This is the frequency duration used when the default value of  $\text{RF} = 1.0$  is used. If  $\text{RF} < 1.0$ , the response between  $\text{RF}$  and 1.0 times  $F$  is set to zero when using the inverse Fourier transform to compute time domain responses.
6. The frequency list used in the frequency response calculations is generated using a constant incremental frequency of  $\text{del } F = \text{RDELTF} * F$ , and the total frequency duration is  $F = \text{RF} * F$ .

Input Data Entry    FLFACT    Aerodynamic Physical Data

Description:    Used to specify density ratios, Mach numbers, and reduced frequencies for flutter analysis.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
FLFACT	SID	F1	F2	F3	F4	F5	F6	F7	CONT
FLFACT	97	.3	.7	3.5					
CONT	F8	F9	etc						

Field

Contents

SID                      Set identification number (Integer > 0).

F1                        Aerodynamic factor (Real).

Remarks:

1. Only the factors selected by a FLUTTER data entry will be used.
2. Imbedded blank fields are forbidden.
3. Parameters must be listed in the order in which they are to be used within the looping of flutter analysis.
4. All FLFACT entries having the same SETID will be treated as a single set.



Input Data Entry FLUTTER Aerodynamic Flutter Data

Description: Defines data needed to perform flutter analysis.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
FLUTTER	SID	METHOD	DENS	MACH	VEL	MLIST	EPS		CONT
FLUTTER	19	PK	119	219	319	10	1.-4		ABC
CONT	SYMxz	SYMxy							
+BC	1	0							

Field

Contents

SID	Set identification number (Integer > 0).
METHOD	Flutter analysis method, "PK" for PK-method,
DENS	Identification number of an FLFACT data entry specifying density ratios to be used in flutter analysis (Integer $\geq 0$ ).
MACH	Identification number of an FLFACT data entry specifying Mach numbers (m) to be used in flutter analysis (Integer > 0 or the Mach No., Real)
VEL	Identification number of an FLFACT data entry specifying the velocities to be used in flutter analysis. (Integer > 0).
MLIST	Identification number of a SET1 entry containing a list of normal modes to be <u>deleted</u> in flutter analysis (Integer $\geq 0$ ).
EPS	Convergence parameter for k; (Real, default = 1.0E-5)
SYMxz, SYMxy	Symmetry flags

Remarks:

1. The FLUTTER data entry must be selected in the solution control packet.
2. The density is given by  $DENS \cdot RHOREF$ , where RHOREF is the reference value given on the AERO data entry, and DENS is the density ratio given in the FLFACT entry.
3. If the MLIST is blank, all the normal modes computed will be retained in the flutter analysis.
4. An eigenvalue is accepted in the PK-method when  $|k - k_{estimate}| < EPS$
5. The symmetry flags will be used to select the appropriate unsteady aerodynamic matrices generated from the list of m-k pairs for each symmetry option given on the MKAERO1 entries.

6. If only one Mach number is desired, it may be placed directly in field 5 of this entry.

Input Data Entry FORCE Static Load

Description: Defines a static load at a grid point by specifying a vector.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
FORCE	SID	G	CID	F	N1	N2	N3		
FORCE	2	5	6	2.9	0.0	1.0	0.0		

Field

Contents

SID	Load set identification number (Integer > 0)
G	Grid point identification number (Integer > 0)
CID	Coordinate system identification number (Integer $\geq$ 0, or blank) (Default = 0)
F	Scale factor (Real)
N1, N2, N3	Components of a vector measured in the coordinate system defined by CID (Real; must have at least one nonzero component)

Remarks:

1. The static load applied to grid point G is given by

$$\{f\} = F \{N\}$$

where {N} is the vector defined in fields 6,7 and 8.

2. A CID of zero references the basic coordinate system.

Input Data Entry FORCE1 Static Load, Alternate Form 1

Description: Used to define a static load by specification of a value and two grid points which determine the direction.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
FORCE1	SID	G	F	G1	G2				
FORCE1	6	13	-2.93	16	13				

Field

Contents

SID Load set identification number (Integer > 0)

G Grid point identification number (Integer > 0)

F Magnitude of load (Real)

G1, G2 Grid point identification numbers (Integer > 0; G1 ≠ G2)

Remarks:

1. The direction of the force is determined by the vector from G1 to G2.

# Input Data Entry FREQ

**Description:** Defines a set of frequencies to be used in the solution of frequency response problems.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
FREQ	SID	F	F	F	F	F	F	F	CONT
FREQ	3	2.98	3.05	17.9	21.3	25.6	28.8	31.2	ABC
CONT	F	F	F	F					
+BC	29.2	22.4	19.3	etc.					

## Field

## Contents

SID Frequency set identification number (Integer > 0)

F Frequency value (Real > 0.0)

## Remarks:

1. The units for the frequencies are cycles per unit time.
2. Frequency sets must be selected by the Solution Control (FSTEP-SID) to be used.
3. All FREQ, FREQ1 and FREQ2 entries with the same frequency set identification numbers will be used. Duplicate frequencies will be ignored.  $f_N$  and  $f_{N-1}$  are considered duplicated if

$$|f_N - f_{N-1}| < 10^{-5} * (f_{MAX} - f_{MIN}).$$

# Input Data Entry FREQ1

Description: Defines a set of frequencies to be used in the solution of frequency response problems by specification of a starting frequency, frequency increment, and number of increments desired.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
FREQ1	SID	F1	DF	NDF					
FREQ1	6	2.9	0.5	13					

## Field

## Contents

SID	Frequency set identification number (Integer > 0)
F1	First frequency in set (Real > 0.0)
DF	Frequency increment (Real > 0.0)
NDF	Number of frequency increments (Integer > 0)

## Remarks:

1. The units for the frequency F1 and the frequency increment DF are cycles per unit time.
2. The frequencies defined by this entry are given by:  

$$f_i = F1 + (i-1) DF, i = 1, NDF + 1$$
3. Frequency sets must be selected by the Solution Control (FSTEP-SID) to be used.
4. All FREQ, FREQ1 and FREQ2 entries with the same frequency set identification numbers will be used. Duplicate frequencies will be ignored.  $f_N$  and  $f_{N-1}$  are considered duplicated if  $|f_N - f_{N-1}| < 10^{-5} * (f_{MAX} - f_{MIN})$ .

# Input Data Entry FREQ2

**Description:** Defines a set of frequencies to be used in the solution of frequency response problems by specification of a starting frequency, final frequency, and number of logarithmic increments desired.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
FREQ2	SID	F1	F2	NF					
FREQ2	6	1.0	8.0	6					

## Field

## Contents

SID	Frequency set identification number (Integer > 0)
F1	First frequency (Real > 0.0)
F2	Last frequency (Real > 0.0; F2 > F1)
NF	Number of logarithmic intervals (Integer > 0)

## Remarks:

- The units for the frequencies F1 and F2 are cycles per unit time.
- The frequencies defined by this entry are given by:  

$$f_i = F1 \cdot e^{(i-1)d}, \quad i = 1, 2, \dots, NF + 1$$

where,  $d = (1/NF) \log_e (F2/F1)$

For the example shown, the list of frequencies will be 1.0, 1.4142, 2.0, 2.8284, 4.0, 5.6569 and 8.0 cycles per unit time.
- Frequency sets must be selected by the Solution Control (FSTEP-SID) to be used.
- All FREQ, FREQ1 and FREQ2 entries with the same frequency set identification numbers will be used. Duplicate frequencies will be ignored.  $f_N$  and  $f_{N+1}$  are considered duplicated if  $|f_N - f_{N+1}| < 10^{-5} * (f_{MAX} - f_{MIN})$

Input Data Entry     FREQLIST

Description:    Defines a list of frequencies for which outputs are desired.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
FREQLIST	SID	FREQ	FREQ	FREQ	FREQ	FREQ	FREQ	FREQ	CONT
FREQLIST	100	10.0	20.0	50.0	100.0				
CONT	FREQ	FREQ	-etc-						

Field

Contents

SID                      Set identification number referenced by Solution Control  
(Integer > 0)

FREQ                    Frequency (in Hertz) at which outputs are desired. (Real)

Remarks:

1. In order to be used, the SID must be referenced by Solution Control.
2. The nearest frequency to FREQ, either above or below, which was used in the Frequency Response analysis will be used to satisfy the output requests.
3. Any number of continuations is allowed.



Input Data Entry    GRAV    Gravity Vector

Description:    Used to define gravity vectors for use in determining gravity loadings for the structural model.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
GRAV	SID	CID	G	N1	N2	N3			
GRAV	1	3	32.2	0.0	0.0	-1.0			

Field

Contents

SID	Set identification number (Integer > 0)
CID	Coordinate system identification number (Integer ≥ 0)
G	Gravity vector scale factor (Real)
N1, N2, N3	Gravity vector components (Real; at least one nonzero component)

Remarks:

1. The gravity vector is defined by  $\{g\} = G(N1, N2, N3)$ . The direction of  $\{g\}$  is the direction of free fall.
2. A CID of zero references the basic coordinate system.
3. Gravity loads may be combined with "simple loads" (e.g., FORCE, MOMENT) by specification on a LOAD entry or by GRAV - SID. Gravity loads with the same SID as simple load entries will not be used unless referenced by one of these methods.
4. Load sets must be selected in Solution Control to be used.
5. The units of G should be length/sec<sup>2</sup> in consistent length units.

Input Data Entry GRDSET Grid Point Default

Description: Defines default options for fields 3, 7, and 8 of all GRID entries.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
GRDSET		CP				CD	PS		
GRDSET		16				32	3456		

Field

Contents

CP	Identification number of coordinate system in which the location of the grid point is defined (Integer $\geq 0$ ).
CD	Identification number of coordinate system in which the displacements are measured at grid points (Integer $\geq 0$ ).
PS	Permanent single-point constraints associated with grid points (any of the digits 1-6 with no imbedded blanks) (Integer $\geq 0$ ).

Remarks:

1. The contents of fields 3, 7, or 8 of this entry are assumed for the corresponding fields of any GRID entry whose field 3, 7, and/or 8 are blank. If any of these fields on the GRID entry is blank, the default option defined by this entry occurs for that field. If no permanent single-point constraints are desired or one of the coordinate systems is basic, the default may be overridden on the GRID entry by making one of the fields 3, 7, or 8 zero (rather than blank). Only one GRDSET entry may appear in the user's Bulk Data Packet.
2. The primary purpose if this entry is to minimize the burden of preparing data for problems with a large amount of repetition (e.g., two-dimensional pinned-joint problems).
3. At least one of the entries CP, CD, or PS must be nonzero.

# Input Data Entry GRID    Grid Point

**Description:** Defines the location of a geometric grid point of the structural model, the directions of its displacement, and its permanent single-point constraints.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
GRID	ID	CP	X1	X2	X3	CD	PS		
GRID	2	3	1.0	2.0	3.0		315		

## Field

## Contents

ID	Grid point identification number (Integer > 0).
CP	Identification number of coordinate system in which the location of the grid point is defined (Integer > 0 or blank)
X1,X2,X3	Location of the grid point in coordinate system CP (Real).
CD	Identification number of coordinate system in which displacements, degrees of freedom, constraints, and solution vectors are defined at the grid point (Integer > 0 or blank)
PS	Permanent single-point constraints associated with grid point (any of the digits 1-6 with no imbedded blanks) (Integer > 0 or blank)

## Remarks:

1. All grid point identification numbers must be unique with respect to all other structural and scalar points.
2. The meaning of X1, X2 and X3 depend on the type of coordinate system, CP, as follows: (see CORDij; entry descriptions)

TYPE	X1	X2	X3
Rectangular	X	Y	Z
Cylindrical	R	$\theta$ (degrees)	Z
Spherical	R	$\theta$ (degrees)	$\phi$ (degrees)

3. The collection of all CD coordinate systems defined on all GRID entries is called the Global Coordinate System. All degrees-of-freedom, constraints, and solution vectors are expressed in the Global Coordinate System.

# Input Data Entry GRIDLIST

Description: Defines a list of points at which outputs are desired.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
GRIDLIST	SID	GID	GID	GID	GID	GID	GID	GID	CONT
GRIDLIST	100	1001	1010	1020					
CONT	GID	GID	-etc-						

## Alternate Form:

1	2	3	4	5	6	7	8	9	10
GRIDLIST	SID	GID	"THRU"	GID					
GRIDLIST	100	100	THRU	200					

## Field

## Contents

SID	Set identification number referenced by Solution Control (Integer > 0 )
GID	Grid, scalar or extra point id at which outputs are desired. (Integer > 0 )

## Remarks:

1. In order to be used, the SID must be referenced by Solution Control.
2. Nonexistent points may be referenced and will result in no error message.
3. Any number of continuations is allowed.
4. Referenced extra points will only appear if the boundary condition includes extra points and then will only appear in Dynamic Response disciplines.

# Input Data Entry GUST Aerodynamic Gust Load Description

**Description:** Defines a stationary vertical gust for use in aeroelastic analysis.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
GUST	SID	GLOAD	WG	XO	V	ODP	MACH		CONT
GUST	133	61	1.0	0.	1.+4	13.5	0.9		ABC
CONT	SYMxz	SYMxy							
+BC	1	0							

## Field

## Contents

SID	Gust set identification number (Integer > 0).
GLOAD	The SID of a TLOAD or RLOAD data entry which defines the time or frequency dependence (Integer > 0).
WG	Scale factor (gust velocity/forward velocity) for gust velocity (Real ≠ 0).
XO	Location of reference plane in aerodynamic coordinates (Real ≥ 0.0).
V	Velocity of vehicle (Real > 0.0).
QDP	Dynamic pressure (Real > 0.0).
MACH	Mach number (Real > 0.0).
SYMxz,SYMxy	Symmetry flags

## Remarks:

1. The GUST entry is selected as a discipline option for FREQUENCY or TRANSIENT in Solution Control.
2. The gust angle is in the +z direction of the aerodynamic coordinate system. The value is

$$WG = T \left[ t - \frac{x - x_0}{V} \right]$$

where T is the tabular function.

3. In random analysis, a unit gust velocity (WG = 1/velocity) is suggested. The actual rms value is entered on the TABRNDG data entry.

4. The symmetry flags will be used to select the appropriate unsteady aerodynamic matrices from the list of m-k pairs for each symmetry option given on the MKAERO1 entries.

# Input Data Entry IC Transient Initial Condition

**Description:** Defines values for the initial conditions of coordinates used in transient analysis. Both displacement and velocity values may be specified at independent coordinates of the model.

## Format and Examples:

	1	2	3	4	5	6	7	8	9	10
IC		SID	G	C	UO	VO				
IC		1	3	2	5.0	-6.0				

## Field

## Contents

SID	Set identification number (Integer > 0)
G	Grid, scalar or extra point identification number (Integer > 0)
C	Component number (blank or zero for scalar or extra points, any <u>one</u> of the digits 1-6 for a grid point)
UO	Initial displacement value (Real)
VO	Initial velocity value (Real)

## Remarks:

1. Transient initial condition sets must be selected in the Solution Control (IC-SID) to be used.
2. If no IC set is selected, all initial conditions are assumed zero.
3. Initial conditions for coordinates not specified on IC entries will be assumed zero.
4. Initial conditions may be used only in direct formulation. In a modal formulation the initial conditions are all zero.

Input Data Entry JSET Select Coordinates for the j-set

Description: Defines degrees of freedom that the user desires to use in the computation of inertia relief mode shape(s) in Dynamic Reduction.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
JSET	SETID	ID	C	ID	C	ID	C		CONT
JSET	16	2	23	3516					
CONT	ID	C	ID	C	-etc-				
JSET									

Field

Contents

SETID The set identification number of the INERTIA set.

ID Grid or scalar point identification number (Integer > 0).

C Component number, zero or blank for scalar points, any unique combination of the digits 1-6 for grid points.

Remarks:

- Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
- When JSET and/or JSET1 entries are present, all degrees of freedom not otherwise constrained will be placed on the o-set.
- Use of JSET in dynamic reduction:
  - JSET defines the structural/non-structural j-set degrees of freedom (inertia relief shapes). An alternate input format is provided by the JSET1 entry.
  - The SID is selected by the Solution Control Command BOUNDARY INERTIA = n.
  - Use "0" as the grid point identification number to select the origin of the basic coordinate system as the j-set degrees of freedom.
- Any number of continuations are allowed.



Input Data Entry **JSET1** Select Coordinates for the j-set, Alternate Form

**Description:** Defines degrees of freedom that the user desires to use in the computation of inertia relief mode shape(s) in Dynamic Reduction.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
JSET1	SETID	C	G	G	G	G	G	G	CONT
JSET1	345	2	1	3	10	9	6		ABC
CONT	G	G	G	etc.					
+bc	7	8		etc.					

**Alternate Form:**

1	2	3	4	5	6	7	8	9	10
JSET1	SETID	C	ID1	"THRU"	ID2				
JSET1	345	12345	7	THRU	109				

#### Field

#### Contents

**SETID** The INERTIA set identification number

**C** Component number (any unique combination of the digits 1-6 (with no imbedded blanks) when point identification numbers are grid points; must be blank or zero if point identification numbers are scalar points.

**G, ID1, ID2** Grid or scalar point identification numbers (Integer > 0, ID1 > ID2).

#### Remarks:

- Coordinates specified on this entry form members of a set that is exclusive from other sets defined by bulk data entries.
- When JSET and/or JSET1 entries are present, all degrees of freedom not otherwise constrained will be placed in the o-set.
- If the alternate form is used, all points in the sequence ID1 through ID2 are required to exist.
- Use of JSET1 in dynamic reduction:
  - JSET1 defines the structural and non-structural j-set degrees of freedom (inertia relief shapes). An alternate input format is provided by the JSET entry.
  - The SID is selected by Solution Control Command BOUNDARY INERTIA = n.

- c. Use "0" as the grid point identification number to select the origin of the basic coordinate system as the j-set degrees freedom.

# Input Data Entry    LOAD    Static Load Combination (Superposition)

**Description:** Defines a static load as a linear combination of load sets defined via FORCE, MOMENT, FORCE1, MOMENT1, PLOAD, and GRAV entries.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
LOAD	SID	S	S1	L1	S2	L2	S3	L3	CONT
LOAD	101	-0.5	1.0	3	6.2	4	4.5	10	ABC
CONT	S4	L4							
+BC	2.3	115	-etc-						

## Field

## Contents

SID                      Load set identification number (Integer > 0)

S                        Scale factor (Real)

S1                       Scale factors (Real)

L1                       Load set identification numbers defined via data entry types enumerated above (Integer > 0)

## Remarks:

1. The load vector defined is given by

$$(P) = S \sum_i S_i (L_i)$$

2. The Li must be unique. The remainder of the physical entry containing the last entry must be blank.
3. Load sets must be selected in the Solution Control if they are to be applied to the structural model.
4. A LOAD entry may not reference a set identification number defined by another LOAD entry.

Input Data Entry **MAT1** Material Property Definition, Form 1

**Description:** Defines the material properties for linear, temperature-independent, isotropic materials

**Format and Example:**

1	2	3	4	5	6	7	8	9	10
MAT1	MID	E	G	NU	RHO	A	TREF	GE	CONT
MAT1	17	3.+7		0.33	4.28	6.5-6	5.37-6	0.23	ABC
CONT	ST	SC	SS						
+BC	20.+4	15.+4	12.+4						

**Field**

**Contents**

MID	Material identification number (Integer >0)
E	Young's modulus (Real or blank)
G	Shear modulus (Real or blank)
NU	Poisson's ratio ( $-1.0 < \text{Real} \leq 0.5$ or blank)
RHO	Mass density (Real $\geq 0.0$ ).
A	Thermal expansion coefficient (Real)
TREF	Thermal expansion reference temperature (Real)
GE	Structural element damping coefficient (Real)
ST, SC, SS	Stress limits for tension, compression, and shear (Real). Used to compute margins of safety in certain elements and to compute stress/strain constraints.

**Remarks:**

1. The material identification number must be unique for all MAT1, MAT2, MAT8 and MAT9 bulk data entries.
2. The mass density, RHO, will be used to automatically compute mass for all structural elements.
3. Weight density may be used in fields 6 if the value 1/g is entered on the CONVERT entry where g is the acceleration of gravity.
4. Either E or G must be specified (i.e., nonblank).
5. If any one of E, G, or NU is blank, it will be computed to satisfy the identity  $E = 2 * (1 + \text{NU}) * G$ ; otherwise, values supplied by the user will be used.

6. If E and NU or G and NU are both blank, they will both be given the values 0.0.
7. Implausible data on one or more MAT1 entries will result in a warning message. Implausible data are defined as any of  $E < 0.0$  or  $G < 0.0$  or  $NU > 0.5$  or  $NU < 0.0$  or  $|1 - E/[2(1+NU)G]| > 0.01$  except for cases covered by Remark 6.
8. It is strongly recommended that only two of the three values E, G, and NU be input. The three values may be input independently on the MAT2 entry.

# Input Data Entry **MAT2** Material Property Definition, Form 2

**Description:** Defines the material properties for linear, temperature-independent, anisotropic materials for two-dimensional elements.

## **Format and Example:**

1	2	3	4	5	6	7	8	9	10
MAT2	MID	G11	G12	G13	G22	G23	G33	RHO	CONT
MAT2	13	6.2+3			6.2+3		5.1+3	0.056	ABC
CONT	A1	A2	A12	TO	GE	ST	SC	SS	
+BC	6.5-6	6.5-6		-500.0	0.002	20.+5			

## **Field**

## **Contents**

MID	Material identification number (Integer > 0)
G <sub>ij</sub>	The material property matrix (Real)
RHO	Mass density (Real ≥ 0.0).
A <sub>i</sub>	Thermal expansion coefficient vector (Real)
TO	Thermal expansion reference temperature (Real)
GE	Structural element damping coefficient (Real)
ST, SC, SS	Stress limits for tension, compression, and shear (Real). Used to compute margins of safety in certain elements; and to compute stress/strain constraints.

## **Remarks:**

1. The material identification numbers must be unique for all MAT1, MAT2, MAT8, and MAT9 bulk data entries.
2. The mass density, RHO, will be used to automatically compute mass for all structural elements.
3. The convention for the G<sub>ij</sub> in fields 3 through 8 are represented by the matrix relationship

$$\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{Bmatrix} = \begin{bmatrix} G_{11} & G_{12} & G_{13} \\ G_{12} & G_{22} & G_{23} \\ G_{13} & G_{23} & G_{33} \end{bmatrix} \begin{Bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_{12} \end{Bmatrix} - (T - T_0) \begin{Bmatrix} A_1 \\ A_2 \\ A_3 \end{Bmatrix}$$

4. 2x2 matrices (for example, transverse shear) use elements G<sub>11</sub>, G<sub>12</sub>, and G<sub>22</sub>. For this case, G<sub>33</sub> must be blank.
5. If the MAT2 entry is referenced by the PCOMP entry, the transverse shear flexibility for the referenced laminae is zero.

6. Unlike the MAT1 entry, data from the MAT2 entry are used directly, without adjustment of equivalent E, G, or NU values.

# Input Data Entry MAT8 Material Property Definition, Form 8

**Description:** Defines the material property for an orthotropic material.

**Format and Example:**

1	2	3	4	5	6	7	8	9	10
MAT8	MID	E1	E2	NU12	G12	G1,Z	G2,Z	RHO	CONT
MAT8	171	30.+6	1.+6	0.3	2.+6	3.+6	1.5+6	0.056	+ABC
CONT	A1	A2	TREF	Xt	Xc	Yt	Yc	SS	CONT
+ABC	28.-6	1.5-6	155.0	1.+4	1.5+4	2.+2	8.+2	1.+3	+DEF
CONT	Ge	F12							
+DEF	1.-4								

## Field

## Contents

MID	Material ID (Integer > 0)
E1	Modulus of elasticity in longitudinal direction (also defined as fiber direction or 1-direction) (Real ≠ 0.0)
E2	Modulus of elasticity in lateral direction (also defined as matrix direction or 2-direction) (Real ≠ 0.0)
NU12	Poisson's ration $[(\epsilon_2)/(\epsilon_1)]$ for uniaxial loading in 1-direction]. Note that $NU21 = (\epsilon_1)/(\epsilon_2)$ for uniaxial loading in 2-direction is related to NU12, E1, E2 by the relation $NU12 \cdot E2 = NU21 \cdot E1$ . (Real)
G12	Inplane shear modulus (Real > 0.0)
G1,Z	Transverse shear modulus for shear in 1-Z plane (Real > 0.0 or blank) (default implies infinity)
G2,Z	Transverse shear modulus for shear in 2-Z plane (Real > 0.0 or blank) (default implies infinity)
RHO	Mass density (Real)
A1	Thermal expansion coefficient in 1-direction (Real)
A2	Thermal expansion coefficient in 2-direction (Real)
TREF	Thermal expansion reference temperature (Real)
Xt, Xc	Allowable stresses in tension and compression, respectively, in the longitudinal direction. Required if failure index is desired or if element stress/strain is constrained in design. (Real ≥ 0.0) (Default value for Xc is Xt)



$Y_t, Y_c$  Allowable stresses in tension and compression, respectively, in the transverse direction. Required if failure index is desired or if the element stress/strain is constrained in design. (Real  $\geq 0.0$ ) (Default value for  $Y_c$  is  $Y_t$ )

SS Allowable stress for inplane shear (Real  $> 0.0$ )

Ge Structural damping coefficient (Real)

$F_{12}$  Interaction term in the tensor polynomial theory of Tsai-Wu (Real). Required if failure index or stress constraint by Tsai-Wu theory is desired and if value of  $F_{12}$  is different from 0.0.

**Remarks:**

1. If  $G_{1,z}$  and  $G_{2,z}$  values are specified as zero, or are not supplied, transverse shear flexibility calculations will not be performed.
2. An approximate value for  $G_{1,z}$  and  $G_{2,z}$  is the in-plane shear modulus  $G_{12}$ . If test data are not available to accurately determine  $G_{1,z}$  and  $G_{2,z}$  for the material and transverse shear calculations are deemed essential, the value of  $G_{12}$  may be supplied for  $G_{1,z}$  and  $G_{2,z}$ .
3.  $X_t, X_c, Y_t, Y_c$  and SS are used for composite element failure calculations when requested in the "F.T." field of the PCOMP<sub>i</sub> entries.

Description: Defines the material properties for linear, temperature-independent, anisotropic materials for solid isoparametric elements.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MAT9	MID	G11	G12	G13	G14	G15	G16	G22	CONT
MAT9	17	6.2+3						6.2+3	ABC
CONT	G23	G24	G25	G26	G33	G34	G35	G36	CONT
+BC									DEF
CONT	G44	G45	G46	G55	G56	G66	RHO	A1	CONT
+EF	5.1+3			5.1+3		5.1+3	3.2	6.6-6	GHI
CONT	A2	A3	A4	A5	A6	TREF	GE		
+HI	6.5-6	6.5-6				125.			

Field

Contents

MID	Material identification number (Integer > 0)
Gij	Elements of the 6x6 symmetric material property matrix (Real)
RHO	Mass density (Real $\geq 0.0$ )
Ai	Thermal expansion coefficient vector (Real)
TREF	Thermal expansion reference temperature (Real)
GE	Structural element damping coefficient (Real)

Remarks:

1. The material identification numbers must be unique for all MAT1, MAT2, MAT8, and MAT9 cards.
2. The mass density RHO will be used to automatically compute mass.
3. Continuation number 4 need not be used.

4. The subscripts 1 through 6 refer to x, y, z, xy, yz, zx, e.g:

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{Bmatrix} = \begin{bmatrix} G_{11} & & & & & \\ G_{12} & G_{22} & & & & \\ G_{13} & G_{23} & G_{33} & & & \\ G_{14} & G_{24} & G_{34} & G_{44} & & \\ G_{15} & G_{25} & G_{35} & G_{45} & G_{55} & \\ G_{16} & G_{26} & G_{36} & G_{46} & G_{56} & G_{66} \end{bmatrix} \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix} = \begin{Bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \end{Bmatrix} (T-T_0)$$

5. To obtain the damping coefficient, GE, multiply the critical damping ratio, C/C<sub>0</sub>, by 2.0.

Input Data Entry MFORM Mass Matrix Form

Description: Defines the form of the mass matrix as consistent (coupled) or lumped.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MFORM	VALUE								
MFORM	LUMPED								

Field

Contents

VALUE A character string denoting the form of the mass matrix. The available forms are

- 1) LUMPED
- 2) COUPLED

Remarks:

1. If more than one MFORM is included in the Bulk Data, any COUPLED value will result in coupled mass being used.
2. If no MFORM is indicated, the LUMPED formulation will be used.

Input Data Entry MKAERO1 Mach Number - Frequency Table

**Description:** Provides a table of Mach numbers (m) and reduced frequencies (k) for unsteady aerodynamic matrix calculation.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
MKAERO1	SYMXZ	SYMXY	m <sub>1</sub>	m <sub>2</sub>	m <sub>3</sub>	m <sub>4</sub>	m <sub>5</sub>	m <sub>6</sub>	CONT
MKAERO1	1	0	0.1	0.7					+ABC
CONT	k <sub>1</sub>	k <sub>2</sub>	k <sub>3</sub>	k <sub>4</sub>	k <sub>5</sub>	k <sub>6</sub>	k <sub>7</sub>	k <sub>8</sub>	
+ABC	.3	.6	1.0						

**Field**

**Contents**

SYMXZ,SYMXY Symmetry flags (Integer). See Remarks 5 and 6.

m<sub>i</sub> List of from 1 to 6 Mach numbers (Real  $\geq$  0.0)

k<sub>j</sub> List of from 1 to 8 reduced frequencies (Real  $\geq$  0.0)

**Remarks:**

1. All combinations of (m,k) will be used.
2. The continuation entry is required.
3. Several MKAEROi entries may be in the input packet. If these data entries are in the packet, they will be used to generate aerodynamic matrices.
4. The first imbedded value of either m or k that is zero for a given entry will be interpreted as zero, a subsequent zero (or blank) will end the table. Values of 0.0 or blank that end either the m list or k list will be ignored. If only a single zero value of either m or k is desired, use the MKAERO2 entry.
5. The symmetry flags have the following definition:
  - +1 for symmetric (not available for SYMXY)
  - 0 for asymmetric
  - 1 for antisymmetric

The m-k pairs generated by this entry will generate aerodynamic matrices having the symmetries selected.

6. m-k pairs may be repeated with different symmetry options.
7. The following are the restrictions associated with the symmetry flags:

- (a) Ground effect (if present) is limited to antisymmetric only. SYMXY = 0 or -1
- (b) Ground effect is not available for supersonic flow.

# Input Data Entry MKAERO2 Mach Number - Frequency Table

Description: Provides a list of Mach numbers (m) and reduced frequencies (k) for aerodynamic matrix calculation.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
MKAERO2	SYMXZ	SYMXY	m <sub>1</sub>	k <sub>1</sub>	m <sub>2</sub>	k <sub>2</sub>	m <sub>3</sub>	k <sub>3</sub>	CONT
MKAERO2	0	0	.10	.60	.70	.30	.70	1.0	ABC
CONT	m <sub>4</sub>	k <sub>4</sub>	m <sub>5</sub>	k <sub>5</sub>	etc.				
+BC	.8	.9	.8	1.0					

## Field

## Contents

SYMYZ, SYMXY Symmetry flags (Integer). See Remarks 4 and 5.

m<sub>1</sub>, k<sub>1</sub> List of pairs of Mach numbers (Real ≥ 0.) and reduced frequencies (Real ≥ 0.)

## Remarks:

1. This entry will cause the aerodynamic matrices to be computed for the given sets of parameters.
2. Several MKAEROi entries may be in the input packet. If these data entries are in the packet, they will be used.
3. Any number of continuations are allowed.
4. The symmetry flags have the following definition:
  - +1 for symmetric (not available for SYMXY)
  - 0 for asymmetric
  - 1 for antisymmetric

The m-k pairs listed on the entry will generate aerodynamic matrices having the symmetries selected.

5. m-k pairs may be repeated with different symmetry options.
6. The following restrictions are imposed on the symmetry flags:
  - (a) Ground effect (if present) must be antisymmetric.

SYMXY = 0 or -1

- (b) Ground effect is not available for supersonic flow.

# Input Data Entry MODELIST

Description: Defines a list of modes at which outputs are desired.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MODELIST	SID	MODE	MODE	MODE	MODE	MODE	MODE	MODE	CONT
MODELIST	100	1	2	4					
CONT	MODE	MODE	-etc-						

Alternate Form:

1	2	3	4	5	6	7	8	9	10
MODELIST	SID	MODE	"THRU"	MODE					
MODELIST	100	1	THRU	10					

## Field

## Contents

SID Set identification number referenced by Solution Control (Integer > 0 )

MODE Mode number of mode at which outputs are desired. (Integer > 0)

## Remarks:

1. In order to be used, the SID must be referenced by Solution Control.
2. Modes are numbered from 1 to n, starting at the lowest frequency for which a eigenvector was computed.
3. Nonexistent modes may be referenced and will result in no error message.
4. Any number of continuations is allowed.



Input Data Entry    MOMENT        Static Moment

Description:    Defines a static moment at a grid point by specifying a vector.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MOMENT	SID	G	CID	M	N1	N2	N3		
MOMENT	2	5	6	2.9	0.0	1.0	0.0		

Field

Contents

SID        Load set identification number (Integer > 0)

G         Grid point identification number (Integer > 0)

CID       Coordinate system identification number (Integer ≥ 0)

M         Scale factor (Real)

N1, N2, N3   Components of vector measured in coordinate system defined by CID  
(Real; at least one nonzero component)

Remarks:

1. The static moment applied to grid point G is given by  
$$(m) = M (N1, N2, N3)$$
2. A CID of zero references the basic coordinate system.

Input Data Entry MOMENT1 Static Moment, Alternate Form 1

Description: Defines a static moment by specification of a value and two grid points which determine the direction.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MOMENT	SID	G	M	G1	G2				
MOMENT	6	13	-2.93	16	13				

Field

Contents

SID Load set identification number (Integer > 0)

G Grid point identification number (Integer > 0)

M Magnitude of moment (Real)

G1, G2 Grid point identification numbers (Integer > 0; G1 ≠ G2)

Remarks:

1. The direction of the moment vector is determined by the vector from G1 and G2.

# Input Data Entry MPC Multipoint Constraint

**Description:** Defines a multipoint constraint equation of the form:

$$\sum_j A_j u_j = 0.0$$

## Format and Example:

1	2	3	4	5	6	7	8	9	10
MPC	SID	G0	C0	A0	G	C	A		CONT
MPC	3	28	3	6.2	2.	3	4.29		+B
CONT		G	C	A	G	C	A		
+B		1	4	-2.91				-etc-	

## Field

## Contents

SID	Set identification (Integer > 0)
G0,G	Identification number of grid or scalar point (Integer > 0)
C0,C	Component number - any <u>one</u> of the digits 1-6 in the case of geometric grid points; blank or zero in the case of scalar points (Integer)
A0,A	Coefficient (Real; A0 must be nonzero)

## Remarks:

1. The first coordinate (G0, C0) in the sequence is assumed to be the dependent coordinate. A dependent degree of freedom assigned by one MPC entry cannot be assigned dependent by another MPC entry.
2. Forces of multipoint constraint are not recovered.
3. Multipoint constraint sets must be selected in Solution Control (MPC = SID) to be used.
4. The m-set coordinates specified on this entry may not be specified on other entries that define mutually exclusive sets.

Input Data Entry MPCADD Multipoint Constraint Set Combination

Description: Defines a multipoint constraint set as a union of multipoint constraint sets defined via MPC entries.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MPCADD	SID	S1	S2	S3	S4	S5	S6	S7	CONT
MPCADD	101	2	3	1	6	4			
CONT	S8	S9	-etc-						

Field

Contents

SID Set identification number (Integer > 0)

S<sub>j</sub> Set identification numbers of multipoint constraint sets defined via MPC entries (Integer > 0)

Remarks:

1. The S<sub>j</sub> must be unique.
2. Multipoint constraint sets must be selected in Solution Control (MPC = SID) to be used.
3. S<sub>j</sub> may not be the identification number of a multipoint constraint set defined by another MPCADD entry.
4. MPCADD entries take precedence over MPC entries. If both have the same set ID, only the MPCADD entry will be used.

# Input Data Entry MPPARM

**Description:** Identify values of user defined optimizer parameters that override the default values.

## Format and Example:

1	2	3	4	5	6	7	8	9	10
MPPARM	PARAM	VALUE	PARAM	VALUE	PARAM	VALUE	PARAM	VALUE	CONT
MPPARM	ISCAL	0	STOL	0.005					
CONT	PARAM	VALUE	-etc-						

## Field

## Contents

PARAM Name of the parameter to be overridden (Text).

VALUE Integer or real value to be used for the parameter.

## Remarks

- Any number of PARAM-VALUE combinations can be specified on an MPPARM entry.
- See EDO software manual (ADS V 1.10) for a definition of parameters, but the most useful are shown below:

REAL PARAMETER	DEFINITION
CT	Constraint tolerance in the Method of Feasible Directions or the Modified Method of Feasible Directions. A constraint is active if its numerical value is more positive than CT.
CTL	Same as CT, but for linear constraints.
CTLMIN	Same as CTMIN, but for linear constraints.
CTMIN	Minimum constraint tolerance for nonlinear constraints. If a constraint is more positive than CTMIN, it is considered to be violated.
DABOBJ	Maximum absolute change in the objective between two consecutive iterations to indicate convergence in optimization.
DABOBM	Absolute convergence criterion for the optimization sub-problem when using sequential minimization techniques.
DABSTR	Same as DABOBJ, but used at the strategy level.

REAL PARAMETER	DEFINITION
DELOBJ	Maximum relative change in the objective between two consecutive iterations to indicate convergence in optimization.
DELOBM	Relative convergence criterion for the optimization sub-problem when using sequential minimization techniques.
DELSTR	Same as DELOBJ, but used at the strategy level.
DLOBJ1	Relative change in the objective function attempted on the first optimization iteration. Used to estimate initial move in the one-dimensional search. Updated as the optimization progresses.
DLOBJ2	Absolute change in the objective function attempted on the first optimization iteration. Used to estimate initial move in the one-dimensional search. Updated as the optimization progresses.
DX1	Maximum relative change in a design variable attempted on the first optimization iteration. Used to estimate the initial move in the one-dimensional search. Updated as the optimization progresses.
DX2	Maximum absolute change in a design variable attempted on the first optimization iteration. Used to estimate the initial move in the one-dimensional search. Updated as the optimization progresses.
EXTRAP	Maximum multiplier on the one-dimensional search parameter, ALPHA in the one-dimensional search using polynomial interpolation/extrapolation.
SCFO	The user-supplied value of the scale factor for the objective function if the default or calculated value is to be overridden.
SCLMIN	Minimum numerical value of any scale factor allowed.
STOL	Tolerance on the components of the calculated search direction to indicate that the Kuhn-Tucker conditions are satisfied.

REAL PARAMETER	DEFINITION
THETAZ	Nominal value of the push-off factor in the Method of Feasible Directions.
XMULT	Multiplier on the move parameter, ALPHA, in the one-dimensional search to find bounds on the solution.
ZRO	Numerical estimate of zero on the computer. Usually the default value is adequate. If a computer with a short word length is used, ZRO = 1.0E-4 may be preferred.

INTEGER PARAMETER	DEFINITION
ISCAL	Scaling parameter. By default, scaling is done every NDV iterations, otherwise scaling is performed every ISCAL iterations.
ITMAX	Maximum number of iterations allowed at the optimizer level.
ITRMOP	The number of consecutive iterations for which the absolute or relative convergence criteria must be met to indicate convergence at the optimizer level.
ITRMST	The number of consecutive iterations for which the absolute or relative convergence criteria must be met to indicate convergence at the strategy level.
JTMAX	Maximum of iterations allowed at the strategy level.

Input Data Entry OMIT Omitted Coordinates

Description: Defines degrees of freedom that the user desires to omit from the problem through matrix partitioning. Used to reduce the number of independent degrees of freedom.

Format and Example:

1	2	3	4	5	6	7	8	9	10
OMIT	SETID	ID	C	ID	C	ID	C		
OMIT	10	16	2	23	3516	54	23		

Field

Contents

SETID            The reduce set identification number (Integer > 0).

ID              Grid or scalar point identification number (Integer > 0).

C                Component number, zero or blank for scalar points, any unique combination of the digits 1-6 for grid points.

Remarks:

1. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
2. In many cases it may be more convenient to use OMIT1, ASET or ASET1 entries.



Input Data Entry OMIT1 Omitted Coordinates, Alternate Form

Description: Defines degrees of freedom that the user desires to omit from the problem through matrix partitioning. Used to reduce the number of independent degrees of freedom.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
OMIT1	SETID	C	G	G	G	G	G	G	CONT
OMIT1	3	2	1	3	10	9	6	5	ABC
CONT	G	G	G	-etc.-					
+BC	7	8		-etc.-					

Alternate Form

1	2	3	4	5	6	7	8	9	10
OMIT1	SETID	C	ID1	"THRU"	ID2				
OMIT1	3	0	17	THRU	109				

Field

Contents

SETID            The reduce set identification number (Integer > 0).

C                Component number (Any unique combination of the digits 1-6 (with no imbedded blanks) when point identification numbers are scalar points).

G, ID1, ID2      Grid or scalar point identification number (Integer > 0; ID1 < ID2).

Remarks:

1. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
2. If the alternate form is used, points in the sequence ID1 through ID2 are required to exist.

Input Data Entry    PAERO1    Aerodynamic Panel Property

Description:    Gives associated bodies for the panels in the unsteady aerodynamic model.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PAERO1	PID	B1	B2	B3	B4	B5	B6		
PAERO1	1	3							

Field

Contents

PID                      Property identification number (referenced by CAERO1) (Integer > 0).

B1,...,B6                ID of CAERO2 entries for associated bodies (Integer  $\geq 0$  or blank).

Remarks:

1. The associated bodies must be in the same aerodynamic group.
2. The Bi numbers above must appear on a CAERO2 entry to define these bodies completely.

# Input Data Entry PAERO2 Aerodynamic Body Properties

Description: Defines the cross-section properties of unsteady aerodynamic bodies.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PAERO2	PID	ORIENT	WIDTH	AR	LRSB	LRIB	LTH1	LTH2	CONT
PAERO2	2	Z	6.0	1.0	22	91	100		ABC

CONT	THI1	THN1	THI2	THN2	THI3	THN3			
+BC	1	3							

## Field

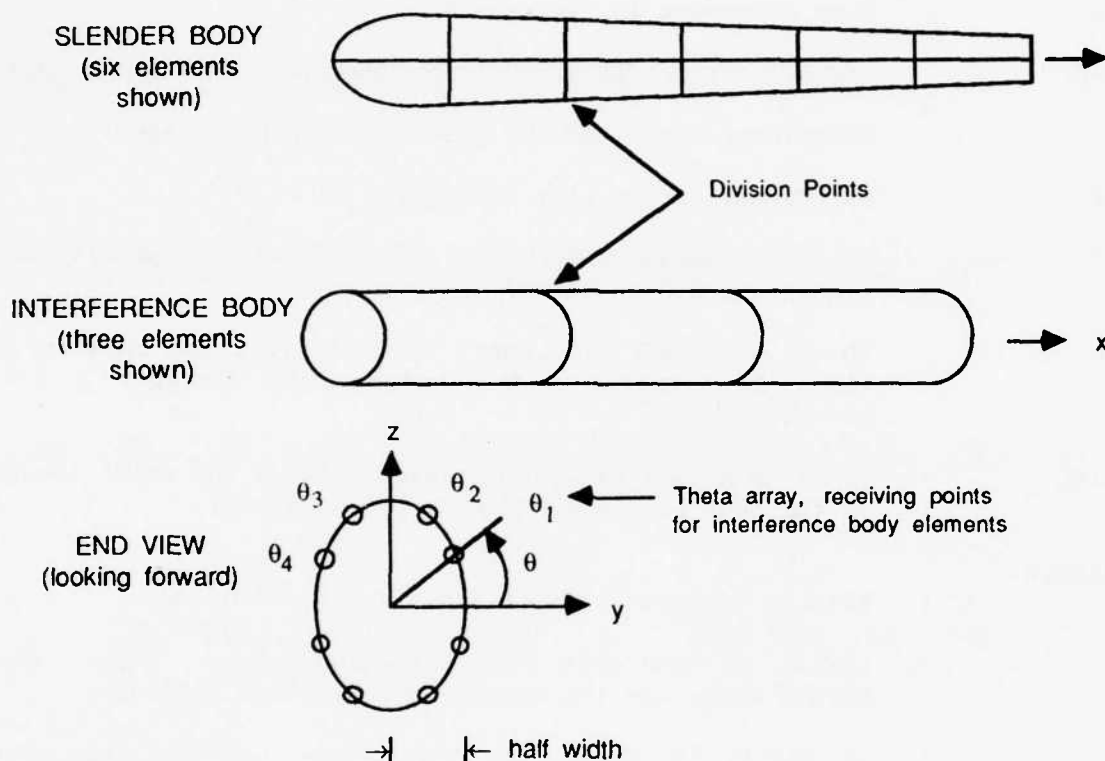
## Contents

PID	Property identification number (Integer > 0).
ORIENT	Orientation flag "Z", "Y", or "ZY". Type of motion allowed for bodies (Text). Refers to the aerodynamic coordinate system "y" and "z" directions (see AERO data entry).
WIDTH	Reference half-width of body (Real > 0.0).
AR	Aspect ratio (height/width) (Real > 0.0).
LRSB	ID of an AEFACT data entry containing a list of slender body half-widths. If blank, the value of WIDTH will be used (Integer > 0 or blank).
LRIB	ID of an AEFACT data entry containing a list of interference body half-widths. If blank, the value of WIDTH will be used (Integer > 0 or blank).
LTH1, LTH2	ID of AEFACT data entries for defining theta arrays for interference calculations (Integer ≥ 0).
THIi, THNi	The first and last interference element of a body to use the theta <sub>i</sub> array (Integer ≥ 0).

## Remarks:

1. The EID of all CAERO2 elements in any IGID group must be ordered so that their corresponding ORIENT values appear in the order Z, ZY, Y.
2. The half-widths (given on AEFACT data entries referenced in fields 6 and 7) are specified at division points. The number of entries on an AEFACT data entry used to specify half-widths must be one greater than the number of elements.
3. The half-width at the first point (i.e., the nose) on a slender body is usually 0.0; thus, it is recommended (but not required) that the LRSB data is supplied with a zero first entry.

4. TH1i and THNi are interference element locations on a body. The element numbering begins at one for each body.
5. A body is represented by a slender body surrounded by an interference body. The slender body creates the downwash due to the motion of the body, while the interference body represents the effects upon panels and other bodies.



Input Data Entry PAERO6

Description: Defines body analysis parameters for steady aerodynamics.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PAERO6	BCID	CMPNT	CP	IGRP	NRAD	LRAD	LAXIAL		
PAERO6	10	FUSEL	0	3	4				

Field

Contents

BCID	Body component ID (Integer > 0)
CMPNT	Component type (FUSEL for the fuselage or POD for a POD)
CP	Coordinate system of the geometric input (Integer)
IGRP	Aerodynamic group flag (Integer > 0)
NRAD	Number of equal radial cuts used to define the body panels (Integer $\geq$ 0.0 or blank)
LRAD	ID of an AEFACT data entry which defines the angular locations in degrees of the body panels (Integer $\geq$ 0.0 or blank).
LAXIAL	ID of an AEFACT data entry which defines the axial locations of the body panels (Integer $\geq$ 0.0 or blank).

Remarks:

1. LRAD is required only if NRAD is zero or blank.
2. LAXIAL is used only for FUSEL components. Inputs on the AEFACT entry are the dimensional fuselage stations.
3. If LAXIAL is blank, the axial locations are the same as those given by AXSTA data entries for the component.

# Input Data Entry PBAR Simple Beam Property

Description: Defines the properties of a simple beam (bar) which is used to create bar elements via the CBAR entry.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PBAR	PID	MID	A	I1	I2	J	NSM	TMIN	CONT
PBAR	39	6	2.9		5.97				123
CONT	C1	C2	D1	D2	E1	E2	F1	F2	CONT
+23			2.0	4.0					
CONT	K1	K2	I12	R12	R22	ALPHA			

## Field

## Contents

PID	Property identification number (Integer > 0).
MID	Material identification number (Integer > 0).
A	Area of bar cross-section (Real).
I1,I2,I12	Area moments of inertia (Real) ( $I_1 \geq 0.0$ , $I_2 \geq 0.0$ , $I_1 I_2 > I_{12}^2$ )
J	Torsional constant (Real).
NSM	Nonstructural mass per unit length (Real).
TMIN	The minimum cross-sectional area in design.
K1,K2	Area factor for shear (Real).
Ci,Di,Ei,Fi	Stress recovery coefficients (Real).
R12,R22,ALPHA	Inertia linking terms for design (see remarks 5 and 6).

## Remarks:

1. PBAR entries may only reference MAT1 material entries.
2. The transverse shear stiffnesses in planes 1 and 2 are  $(K1)AG$  and  $(K2)AG$ , respectively. The default values for K1 and K2 are infinite; in other words, the transverse shear flexibilities are set equal to zero. K1 and K2 are ignored if  $I_{12} = 0$ .
3. The stress recovery coefficients C1 and C2, etc., are the y and z coordinates in the BAR element coordinate system of a point at which stresses are computed. Stresses are computed at both ends of the BAR.

4. The TMIN value is used only for shape function design variable linking.
5. For design, the following applies to the R12 and R22 values. The moments of inertia are linked to the cross-sectional area by the following expressions:

$$I_1 = R12 * A^{ALPHA}$$

$$I_2 = R22 * A^{ALPHA}$$

- (A) If R12 = 0.0 then the missing value is computed from  $R12 = I_1 / (A^{ALPHA})$ . The same is true for R22 and I2.
  - (B) The ALPHA value defaults to 1.0 and must be  $\geq 1.0$ .
  - (C) If both I1 and R12 or I2 and R22 are given, the linking expression will override the input I<sub>i</sub> values.
6. If the CBAR is to be designed, the following restrictions apply.
    - (A)  $J = NSM = K1 = K2 = I12 = 0.0$   
If any of these values are not zero, a warning message will be issued and the value set to zero.

# Input Data Entry PCOMP Layered Composite Element Property

Description: Defines the properties of an n-ply composite material laminate.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PCOMP	PID	Z0	NSM	SBOND	F.T.	TMIN		LOPT	CONT
PCOMP	100	-0.5	1.5	5.+3	HOFF			MEM	+ABC
CONT	MID1	T1	TH1	SOUT1	MID2	T2	TH2	SOUT2	CONT
+BC	150	0.05	90.	YES			-45.		+DEF
CONT	MID3	T3	TH3	SOUT3	etc...				
+EF			45.0		...				

## Field

## Contents

PID	Property ID (Integer > 0).
Z0	Offset of the element mid-plane from the bottom surface (Real or blank, see Remark 2)
NSM	Non-structural mass per unit area (Real).
SBOND	Allowable shear stress of the bonding material. (Real $\geq$ 0.0)
F.T.	Failure theory, one of the strings "HILL", "HOFF", "TSAI", "STRESS", or "STRAIN". See Remark 4.
TMIN	Minimum ply thickness for design (Real > 0.0 or blank).
LOPT	Lamination generation option, either the string "MEM" or blank. See Remark 5.
MID <sub>i</sub>	Material identification number of the i(th) layer. (Integer > 0 or blank)
T <sub>i</sub>	Thickness of the i(th) layer (Real, > 0.0 or blank).
TH <sub>i</sub>	Angle between the longitudinal direction of the fibers of the i(th) layer and the material X-axis. (Real or blank)
SOUT <sub>i</sub>	Stress output request for i(th) layer, one of the strings "YES" or "NO". (Default = "NO")

## Remarks:

1. For non-designed elements, the plies are numbered from 1 to n beginning with the bottom layer.
2. The default for Z0 is half the laminate thickness.



3. SBOND is required if bonding material failure index calculations are desired.
4. The failure theory is used to determine the element failure on a ply-by-ply basis. The available theories are:
  - HILL - Hill Theory
  - HOFF - Hoffman Theory
  - TSAI - Tsai-Wu Theory
  - STRESS - For Maximum Stress Theory
  - STRAIN - For Maximum Strain Theory
5. MEM indicates a layup of membrane only plies.
6. The material properties, MIDi, may reference only MAT1, MAT2, and MAT8 Bulk Data entries.
7. If any of the MIDi, Ti or THi are blank, then the last non-blank values specified for each will be used to define the values for the ply.
8. TMIN will be ignored unless the element is linked to design variables by ELIST entries.

# Input Data Entry PCOMPL Layered Composite Element Property

**Description:** Defines the properties of an n-ply laminated composite material where all plies are composed of the same material and are of equal thickness.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PCOMPL	PID	Z0	NSM	SBOND	F.T.	TMIN	MID	LOPT	CONT
PCOMPL	100	-0.5	1.7	5.+3	STRAIN		200		+ABC
CONT	TPLY	TH1	TH2	TH3	TH4	TH5	TH6	TH7	CONT
+BC	0.25	-45.0	45.0	90.0	90.0	45.0			
CONT	TH8	TH9	TH10	-etc-					

## Field

## Contents

PID	Property ID (Integer > 0).
Z0	Offset of the element mid-plane from the lower surface (Real or blank, see Remark 2)
NSM	Non-structural mass per unit area (Real).
SBOND	Allowable shear stress of the bonding material. (Real $\geq$ 0.0)
F.T.	Failure theory, one of the strings "HILL", "HOFF", "TSAI", "STRESS", or "STRAIN". See Remark 4.
TMIN	Minimum ply thickness for design (Real > 0.0 or blank).
MID	Material identification number for all layers. (Integer > 0 or blank)
LOPT	Lamination generation option, either the string "MEM" or blank. See Remark 5.
TPLY	Thickness of all layers. (Real > 0.0).
TH <sub>i</sub>	Angle between the longitudinal direction of the fibers of the i(th) layer and the material X-axis. (Real or blank)

## Remarks:

1. For non-designed elements, the plies are numbered from 1 to n beginning with the bottom layer.
2. The default for Z0 is half the laminate thickness.
3. SBOND is required if bonding material failure index calculations are desired.

4. The failure theory is used to determine the element failure on a ply-by-ply basis. The available theories are:

- HILL - Hill Theory
- HOFF - Hoffman Theory
- TSAI - Tsai-Wu Theory
- STRESS - For Maximum Stress Theory
- STRAIN - For Maximum Strain Theory

5. MEM indicates a layup of membrane only plies.
6. The material property, MID, may reference only MAT1, MAT2, or MAT8 Bulk Data entries.
7. TMIN will be ignored unless the element is linked to design variables by ELIST entries.

# Input Data Entry    PCOMP2            Layered Composite Element Property

Description:       Defines the properties of an n-ply laminated composite material where all plies are composed of the same material but are of different thicknesses.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PCOMP2	PID	Z0	NSM	SBOND	F.T.	TMIN	MID	LOPT	CONT
PCOMP2	100	-0.5	1.7	5,+3	TSAI		200		+ABC
+BC	T1	TH1	T2	TH2	T3	TH3	ETC...		
+BC	0.25	-45.0	0.5	90.0	0.25	45.0	.....		

<u>Field</u>	<u>Contents</u>
PID	Property ID (Integer > 0).
Z0	Offset of the element mid-plane from the lower surface (Real or blank, see Remark 2)
NSM	Non-structural mass per unit area (Real).
SBOND	Allowable shear stress of the bonding material. (Real $\geq$ 0.0)
F.T.	Failure theory, one of the strings "HILL", "HOFF", "TSAI", "STRESS", or "STRAIN". See Remark 4.
TMIN	Minimum ply thickness for design (Real > 0.0 or blank).
MID	Material identification number for all layers. (Integer > 0.0 or blank)
LOPT	Lamination generation option, either the string "MEM" or blank. See Remark 5.
T <sub>i</sub>	Thickness of the i(th) layer. (Real > 0.0 or blank).
TH <sub>i</sub>	Angle between the longitudinal direction of the fibers of the i(th) layer and the material X-axis. (Real or blank)

## Remarks:

1. For non-designed elements, the plies are numbered from 1 to n beginning with the bottom layer.
2. The default for Z0 is half the laminate thickness.
3. SBOND is required if bonding material failure index calculations are desired.

4. The failure theory is used to determine the element failure on a ply-by-ply basis. The available theories are:

- HILL - Hill Theory
- HOFF - Hoffman Theory
- TSAI - Tsai-Wu Theory
- STRESS - For Maximum Stress Theory
- STRAIN - For Maximum Strain Theory

5. MEM indicates a layup of membrane only plies.
6. The material property, MID, may reference only MAT1, MAT2, or MAT8 Bulk Data entries.
7. If any of the T1 or T2 are blank, then the last non-blank values specified for each will be used to define the values for the ply.
8. TMIN will be ignored unless the element is linked to design variables by ELIST entries.

Input Data Entry PELAS Scalar Elastic Property

Description: Used to define the stiffness, damping coefficient, and stress coefficient of a scalar elastic element (spring) by means of the CELAS1 entry.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PELAS	PID	K	GE	S	TMIN				
PELAS	7	4.29	0.06	7.92					

Field

Contents

PID	Property identification number (Integer > 0)
K	Elastic property value (Real)
GE	Damping coefficient (Real)
S	Stress coefficient (Real)
TMIN	Minimum value for design

Remarks:

1. The user is cautioned to be careful using negative spring values.
2. TMIN is ignored unless the element is designed using shape function design variable linking.

# Input Data Entry **PIHEX** Isoparametric Hexahedron Property

**Description:** Defines the properties of an isoparametric solid element, including a material reference and the number of integration points. Referenced by the CIHEX1, CIHEX2, and CIHEX3 entries.

## Format and Examples:

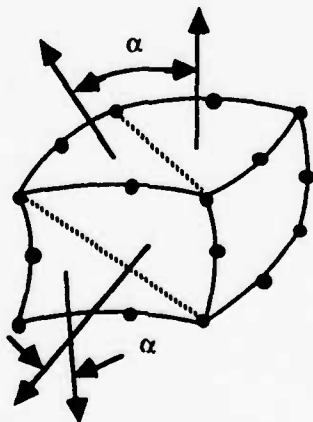
1	2	3	4	5	6	7	8	9	10
PIHEX	PID	MID	CID	NIP	AR	ALFA	BETA		
PIHEX	15	3		3			5.0		

## Field

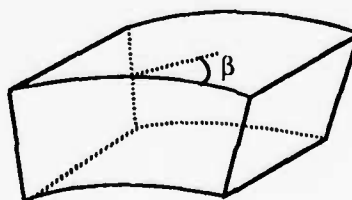
## Contents

PID	Property identification number (Integer > 0)
MID	Material identification number (Integer > 0)
CID	Identification number of the coordinate system in which the material referenced by MID is defined (Integer $\geq 0$ or blank)
NIP	Number of integration points along each edge of the element (Integer = 2, 3, 4, or blank)
AR	Maximum aspect ratio (ratio of longest to shortest edge) of the element (Real > 1.0 or blank)
ALFA	Maximum angle in degrees between the normals of two subtriangles comprising a quadrilateral face (Real, $0.0 \leq \text{ALFA} \leq 180.0$ or blank)
BETA	Maximum angle in degrees between the vector connecting a corner point to an adjacent midside point and the vector connecting that midside point and the other midside or corner point (Real, $0.0 \leq \text{BETA} \leq 180.0$ or blank)

## Examples of Field Definitions:



Example of ALFA



Example of BETA

Remarks:

1. All PIHEX cards must have unique identification numbers.
2. CID is not used for isotropic materials.
3. The default for CID is the basic coordinate system.
4. The default for NIP is 2 for IHEX1 and 3 for IHEX2 and IHEX3.
5. AR, ALFA, and BETA are used for checking the geometry of the element. The defaults are:

	AR	ALFA (degrees)	BETA (degrees)
CIHEX1	5.0	45.0	----
CIHEX2	10.0	45.0	45.0
CIHEX3	15.0	45.0	45.0



## Input Data Entry PLIST

Description: Defines property entries associated with a design variable.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
PLIST	DVID	PTYPE	PID1	PID2	PID3	PID4	PID5	PID6	CONT
PLIST	6	PROD	12	14	22				
CONT	PID7	PID8	PID9	-etc-					

### Alternate Form:

PLIST	DVID	PTYPE	PID1	THRU	PID2				
PLIST	25	PROD	8	THRU	25				

### Field

### Contents

DVID                      Property list identifier (Integer).

PTYPE                     Property type associated with this list (e.g., PROD).

PID1, PID2,  
PID3                      Property entry identifications.

### Remarks:

1. Allowable PTYPES are: PROD, PSHEAR, PCOMP, PCOMP1, PCOMP2, PSHELL, PMASS, PELAS, PTRMEM, PQDMEM1, and PBAR.
2. If the alternate form is used, PID2 must be greater than or equal to PID1.
3. All elements using properties listed as PLIST entries for a particular DVID, will be designed by (linked to) that design variable.

# Input Data Entry PLOAD Static Pressure Load

Description: Defines a static pressure load on a triangular or quadrilateral element.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PLOAD	SID	P	G1	G2	G3	G4			
PLOAD	1	-4.0	16	32	11				

## Field

## Contents

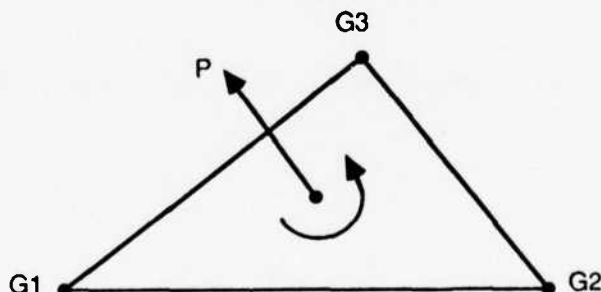
SID Load set identification number (Integer > 0)

P Pressure (Real)

G1,...,G4 Grid point identification numbers (Integer > 0; G4 may be zero or blank)

## Remarks:

1. The grid points define either a triangular or a quadrilateral surface to which a pressure is applied. If G4 is zero or blank, the surface is triangular.
2. In the case of a triangular surface, the assumed direction of the pressure is computed according to the right-hand rule using the sequence of grid points G1, G2, and G3 as illustrated below.



The total load on the surface, AP, is divided into three equal parts and applied to the grid points as concentrated loads. A minus sign in field 3 reverses the direction of the load.

3. In the case of a quadrilateral surface, the grid points G1, G2, G3, and G4 should form a consecutive sequence around the perimeter. The right-hand rule is applied to find the assumed direction of the pressure. Four concentrated loads are applied to the grid points in approximately the same manner as for a triangular surface. The following specific

procedures are adopted to accommodate irregular and/or warped surfaces:

- a. The surface is divided into two sets of overlapping triangular surfaces. Each triangular surface is bounded by two of the sides and one of the diagonals of the quadrilateral.
- b. One-half of the pressure is applied to each triangle which is then treated in the manner described in Remark 2.

Input Data Entry PMASS Scalar Mass Property

Description: Used to define the mass value of a scalar mass element which is defined by means of the CMASS1 entries.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PMASS	PID	M	TMIN	PID	M	TMIN			
PMASS	7	4.29	0.2	6	13.2	0.1			

Field

Contents

PID                      Property identification number (Integer > 0).

M                        Value of scalar mass (Real).

TMIN                     The minimum mass value in design. Default = 0.0001

Remarks:

1. This entry defines a mass value.
2. Up to two mass values may be defined by this entry.
3. TMIN is ignored unless the mass element is linked to design variables through ELIST entries.

Input Data Entry PODMEM1 Quadrilateral Membrane Property

**Description:** Used to define the properties of a quadrilateral membrane referenced by the CQDMEM1 entry. No bending properties are included.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
PODMEM	PID	MID	T	NSM	TMIN				
POMEM	235	2	0.5	0.0					

**Field**

**Contents**

PID                      Property identification number (Integer > 0).

MID                      Material identification number (Integer > 0).

T                        Thickness of membrane (Real > 0.0)

NSM                      Nonstructural mass per unit area (Real).

TMIN                     Minimum thickness for design (Real > 0.0 or blank). Default  
= 0.0001.

**Remarks:**

1. All PQDMEM1 entries must have unique property identification numbers.
2. TMIN is ignored unless the element is linked to the global design variable by ELIST entries.

# Input Data Entry PROD Rod Property

Description: Defines the properties of a rod which is referenced by the CROD entry.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
PROD	PID	MID	A	J	C	NSM	TMIN		
PROD	17	23	42.6	17.92	4.236	0.5			

## Field

## Contents

PID	Property identification number (Integer > 0).
MID	Material identification number (Integer > 0).
A	Area of rod (Real).
J	Torsional constant (Real).
C	Coefficient to determine torsional stress (Real)
NSM	Nonstructural mass per unit length (Real).
TMIN	Minimum rod area for design (Real > 0.0 or blank). Default - 0.0001

## Remarks:

1. PROD entries must all have unique property identification numbers.
2. For structural problems, PROD entries may only reference MAT1 material entries.
3. The formula used to compute torsional stress is:

$$\tau = \frac{cM_{\theta}}{J}$$

where  $M_{\theta}$  is the torsional moment.

4. TMIN is ignored unless the rod element is linked to the design variables by ELIST entries.

Input Data Entry PSHEAR Shear Panel Property

**Description:** Defines the elastic properties of a shear panel. Referenced by the CSHEAR entry.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
PSHEAR	PID	MID	T	NSM	TMIN				
PSHEAR	13	2	4.9	16.2					

**Field**

**Contents**

PID	Property identification number (Integer > 0).
MID	Material identification number (Integer > 0).
T	Thickness of shear panel (Real = 0.0)
NSM	Nonstructural mass per unit area (Real).
TMIN	Minimum panel thickness for design (Real > 0.0 or blank). Default = 0.0001

**Remarks:**

1. All PSHEAR entries must have unique identification numbers.
2. PSHEAR entries may reference only MAT1 material entries.
3. TMIN is ignored unless the element is linked to global design variables by ELIST entries.

Input Data Entry PSHELL Shell Element Property

Description: Defines the membrane, bending, transverse shear, and coupling properties of the shell elements. (QUAD4)

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PSHELL	PID	MID1	T	MID2	12I/T <sup>3</sup>	MID3	TS/T	NSM	CONT
PSHELL	203	204	1.90	205	1.2	206	0.8	6.32	ABC
CONT	Z1	Z2	MID4	MCSID	SCSID	Z0	TMIN		
+BC	+.95	-.95		0	0	0.01			

Field

Contents

PID	Property identification number (Integer > 0).
MID1	Material identification number for membrane (Integer > 0 or blank).
T	Default value for membrane thickness (Real).
MID2	Material identification number for bending (Integer > 0 or blank)
12I/T <sup>3</sup>	Bending stiffness parameter (Real or blank, default = 1.0).
MID3	Material identification number for transverse shear (Integer > 0 or blank, must be blank unless MID2 > 0).
TS/T	Transverse shear thickness divided by membrane thickness (Real or blank, default = .833333).
NSM	Nonstructural mass per unit area (Real).
Z1,Z2	Fiber distances for stress computation. The positive direction is determined by the righthand rule and the order in which the grid points are listed on the connection entry. (Real or blank, defaults are -1/2 T for Z1 and 1/2 T for Z2.)
MID4	Material identification number for membrane-bending coupling (Integer > 0 or blank, must be blank unless MID1 > 0 and MID2 > 0, may not equal MID1 or MID2).
MCSID	Identification number of material coordinate system (Real or blank, or Integer ≥ 0) (See Remark 9)
SCSID	Identification number of stress coordinate system (Real or blank, or Integer ≥ 0) (See Remark 9)



ZO                    Offset of the mid reference plane from the plane of grid points. (Real or blank, default = 0.0) (See Remark 10)

TMIN                Minimum thickness for design (Real > 0.0 or blank). Default = 0.0001

**Remarks:**

1. All PSHELL property entries must have unique identification numbers.
2. The structural mass is computed from the density using the membrane material properties.
3. The results of leaving an MID field blank are:  
MID1    No membrane or coupling stiffness.  
MID2    No bending, coupling, or transverse shear stiffness.  
MID3    No transverse shear flexibility.  
MID4    No bending-membrane coupling.
4. The continuation entry is not required.
5. The MID4 field should be left blank if the material properties are symmetric with respect to the middle surface of the shell.
6. This entry is used only with the CQUAD4 elements.
7. PSHELL entries may reference MAT1, MAT2, or MAT8 material property entries.
8. If the transverse shear material, MID3, references MAT2 data, then G33 must be zero. If MID3 references MAT8 data, then G1, Z and G2, Z must not be zero.
9. If MCSID/SCSID is left blank (0.0) or is real, it is considered to be the angle of rotation of the X axis of the material/stress coordinate system with respect to the X axis of the element coordinate system in the XY plane of the latter. If integer, the orientation of the material/stress x-axis is along the projection of the x-axis of the specified coordinate system onto the x-y plane of the element system. The value of MCSID is the default value for the TM field on CQUAD4 Bulk Data entries.
10. The value of ZO is the default value for the corresponding field on CQUAD4 Bulk Data entries. The default is half the thickness.
11. TMIN is ignored unless element is linked to global design variables by ELIST entries.

Input Data Entry PTRMEM

Description: Defines property data for TRMEM element.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
PTRMEM	PID	MID	T	NSM	TMIN				
PTRMEM	500	1000	0.15						

Field

Contents

PID	Property entry identification number (Integer > 0).
MID	Material property identification (Integer > 0).
T	Thickness of membrane element (Real > 0.0).
NSM	Nonstructural mass associated with the element.
TMIN	Minimum thickness for design (Real > 0.0 or blank). Default - 0.0001

Remarks:

1. The PTRMEM entry can reference either MAT1, MAT2 or MAT8 entries.
2. TMIN is ignored unless the element is linked to global design variables by ELIST entries.

# Input Data Entry RLOAD1

Description: Defines a frequency dependent dynamic load of the form.

$$P(f) = (A[C(f) + iD(f)] e^{i(\theta - 2\pi fr)})$$

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
RLOAD1	SID	ILAG	TC	TD					
RLOAD1	10	3	1	2					

## Field

## Contents

SID	Set identification number (Integer > 0)
ILAG	Identification number of a DLAGS set which defines A, $\theta$ and $r$ (Integer > 0)
TC	Set identification number of TABLEDi entry which gives $C(f)$ (Integer $\geq 0$ ; TC + TD > 0)
TD	Set identification number of TABLEDi entry which gives $D(f)$ (Integer $\geq 0$ ; TC + TD > 0)

## Remarks:

1. RLOAD1 loads may be combined with RLOAD2 loads only by specification on a DLOAD entry.
2. SID must be unique for all RLOAD1, RLOAD2, TLOAD1 and TLOAD2 entries.

# Input Data Entry RLOAD2

Description: Defines a frequency dependent dynamic load of the form.

$$P(f) = (AB(f)e^{i(\phi(f) + \theta - 2\pi fr)})$$

Format and Examples:

1	2	3	4	5	6	7	8	9	10
RLOAD2	SID	ILAG	TB	TP					
RLOAD2	10	6	100	101					

## Field

## Contents

SID	Set identification number (Integer > 0)
ILAG	Identification of a DLAGS entry which defines A, $\theta$ and $r$ (Integer > 0)
TB	Set identification number of TABLEDi entry which gives B(f) (Integer > 0)
TP	Set identification number of TABLEDi entry which gives $\phi(f)$ in degrees (Integer $\geq 0$ )

## Remarks:

1. RLOAD2 loads may be combined with RLOAD1 loads only by specification on a DLOAD entry.
2. SID must be unique for all RLOAD1, RLOAD2, TLOAD1 and TLOAD2 entries.

Input Data Entry SAVE

Description: Defines a list of data base entities that are not to be purged.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
SAVE	NAME1	NAME2	NAME3	NAME4	NAME5	NAME6	NAME7	NAME8	CONT
SAVE	DVCT								
CONT	NAME9	NAME10	NAME11	-etc.-					

Field

Contents

NAME1                      The name of a data base entity whose contents are not to be purged.

Remarks:

1. Any number of continuations are allowed.
2. These data are used by the UTPURG utility to determine if a requested purge of an entity will take place.

# Input Data Entry SEOGP Grid and Scalar Point Resequencing

**Description:** Used to manually order the grid points and scalar points of the problem. The purpose of this entry is to allow the user to reidentify the formation sequence of the grid and scalar points of the structural model in such a way as to optimize bandwidth.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
SEOGP	ID	SEQID	ID	SEQID	ID	SEQID	ID	SEQID	CONT
SEOGP	5392	15.6	596	0.2	2	1.9.2.6	3		
CONT	ID	SEQID	-etc.-						

## Field

## Contents

ID                      Grid point identification number (Integer > 0 )

SEQID                   Sequenced identification number (a special number described below.

## Remarks:

1. ID is any grid or scalar point identification number which is to be reidentified for sequencing purposes. The sequence number is a special number which may have any of the following forms where X is a decimal integer digit - XXXX.X.X.X, XXXX.X.X, XXXX.X or XXXX where any of the leading X's may be omitted. This number must contain no imbedded blanks. The leading character must not be a decimal point.
2. If the user wishes to insert a point between two already existing grid or scalar points, such as 15 and 16, for example, he would define it as, say 5392, and then use this entry to insert extra point number 5392 between them by equivalencing it to, say, 15.6. All output referencing this point will refer to 5392.
3. The SEQID numbers must be unique and may not be the same as a point ID which is not being changed. No extra point ID may be referenced more than once.
4. If a point ID is referenced more than once, the last reference will determine its sequence.

Input Data Entry SET1 Set definition for aerodynamic analyses.

Description: Defines a set of integers by a list.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
SET1	SID	G1	G2	G3	G4	G5	G6	G7	CONT
SET1	3	31	62	93	124	16	17	18	ABC
CONT	G8	-etc.-							
+BC	19								

Alternate Form:

1	2	3	4	5	6	7	8	9	10
SET1	SID	G1	THRU	G2					
SET1	3	5	THRU	10					

Field

Contents

SID Set of identification numbers (Integer > 0)

G1,G2,etc. List of integers (Integer > 0)

Remarks:

1. These entries are referenced by the SPLINE1 and FLUTTER data entries.
2. When using the "THRU" option, all intermediate points must exist.
3. When used by SPLINE1, the entry refers to a list of structural grid points.
4. When used by FLUTTER, the entry refers to mode numbers to be omitted in the flutter analysis.

# Input Data Entry SET2 Grid Point List

Description: Defines a set of structural grid points in terms of aerodynamic macroelements.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
SET2	SID	SP1	SP2	CH1	CH2	ZMAX	ZMIN		
SET2	3	0	.73	0	.667	1.0	-3.51		

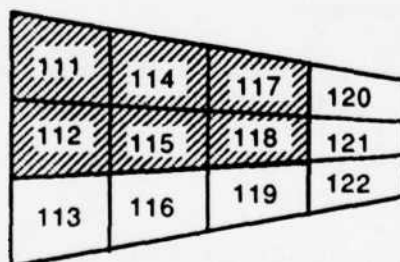
## Field

## Contents

SID	Set identification number (Integer > 0).
SP1,SP2	Lower and higher fractional span division points defining prism containing the set (Real > -.01 < 1.0).
CH1,CH2	Lower and higher fractional chord division points defining prism containing the set (Real > -.01 < 1.01).
ZMAX,ZMIN	Z-coordinates of top and bottom (using right-hand rule with the order the corners as listed on a CAEROi entry) of the prism containing the set (Real). Usually ZMAX > 0.0, ZMIN < 0.0.

## Remarks:

1. These entries are referenced by the SPLINE1 data entries.
2. Every grid point, within the defined prism and within the height range, will be in the set. For example,



The shaded area in the figure defines the cross-section of the prism for the sample data given above. Points exactly on the boundary may be missed; hence, to get all the grid points within the area of the macro element, use SP1 = -.01, SP2 = 1.01, etc.

3. A zero value for ZMAX or ZMIN implies infinity is to be used.



**Input Data Entry SPC****Single-Point Constraint**

**Description:** Defines sets of single-point constraints and enforced displacements.

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
SPC	SID	G	C	D	G	C	D		
SPC	2	32	436	-2.6	5		+2.9		

**Field****Contents**

SID	Identification number of single-point constraint set (Integer > 0)
G	Grid or scalar point identification number (Integer > 0)
C	Component number of global coordinate ( $6 \geq \text{Integer} \geq 0$ ; up to six unique digits may be placed in the field with no imbedded blanks).
D	Value of enforced displacement for all coordinates defined by G and C (Real)

**Remarks:**

1. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
2. Single-point forces of constraint may be recovered during data recovery.
3. Single-point constraint sets must be selected in Solution Control (SPC - SID) to be used.
4. From one to twelve single-point constraints may be defined on a single entry.
5. SPC degrees of freedom may be redundantly specified as permanent constraints on the GRID entry.
6. Continuation entries are not allowed.

Input Data Entry SPCADD Single-Point Constraint Set Combination

Description: Defines a single-point constraint set as a union of single-point constraint sets defined via SPC or SPC1 entries.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
SPCADD	SID	S1	S2	S3	S4	S5	26	S7	CONT
SPCADD	101	3	2	9	1				
CONT	S8	S9	-etc.-						

Field

Contents

SID Identification number for new single-point constraint set (Integer > 0)

Si Identification numbers of single-point constraint sets defined via SPC or by SPC1 entries (Integer > 0; SID ≠ Si)

Remarks:

1. Single-point constraint sets must be selected in Solution Control (SPC = SID) to be used.
2. No Si may be the identification number of a single-point constraint set defined by another SPCADD entry.
3. The Si values must be unique.
4. SPCADD entries take precedence over SPC or SPC1 entries. If both have the same set ID, only the SPCADD entry will be used.

# Input Data Entry SPC1

## Single-Point Constraint, Alternate Form 1

**Description:** Defines sets of single-point constraints

**Format and Examples:**

1	2	3	4	5	6	7	8	9	10
SPC1	SID	C	G1	G2	G3	G4	G5	G6	CONT
SPC1	3	2	1	3	10	9	6	5	ABC
CONT	G7	G8	G9	-etc.-					
+BC	2	8							

**Alternate Form**

1	2	3	4	5	6	7	8	9	10
SPC1	SID	C	GID1	THRU	GID2				
SPC1	3	456	10	THRU	1000				

### Field

### Contents

SID	Identification number of single-point constraint set (Integer > 0)
C	Component number of global coordinate: any unique combination of the digits 1-6 (with no imbedded blanks) when point identification numbers are grid points; must be blank or zero if point identification numbers are scalar points.
G1,GID1	Grid or scalar point identification numbers (Integer > 0)

### Remarks:

1. Note that enforced displacements are not available via this entry. As many continuation entries as desired may appear.
2. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
3. Single-point constraint sets must be selected in Solution Control (SPC = SID) to be used.
4. SPC degrees of freedom may be redundantly specified as permanent constraints on the GRID entry.
5. If the alternate form is used, points in the sequence GID1 through GID2 are required to exist.

Input Data Entry SPLINE1 Surface Spline

Description: Defines a surface spline for interpolating out-of-plane motion for aeroelastic problems.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
<u>SPLINE1</u>	<u>EID</u>	<u>CP</u>	<u>CAERO</u>	<u>BOX1</u>	<u>BOX2</u>	<u>SETG</u>	<u>DZ</u>		
SPLINE1	3		111	111	118	14	0.0		

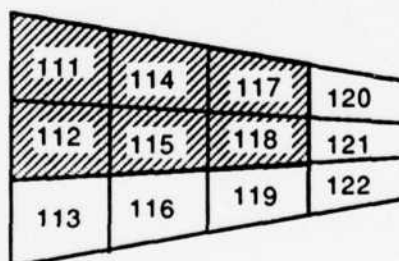
Field

Contents

EID	Element identification number (Integer > 0)
CP	Coordinate system defining the spline plane
CAERO	Aero element ID which defines plane of spline (Integer > 0)
BOX1, BOX2	First and last box whose motions are interpolated using this spline (Integer > 0, BOX2 > BOX1)
SETG	Refers to a SETi entry which lists the structural grid points to which the spline is attached (Integer > 0)
DZ	Linear attachment flexibility (Real $\geq 0.0$ )

Remarks:

1. The interpolated points (k-set) will be defined by aero-cells. The sketch shows the cells for which  $u_k$  is interpolated if BOX1 = 111 and BOX2 = 118.



2. The attachment flexibility (units of area) is used for smoothing the interpolation. If  $DZ = 0.0$ , the spline will pass through all deflected grid points. If  $DZ \gg$  (area of spline), a least squares plane fit will occur. Intermediate values will provide smoothing.
3. If no CP is specified, the spline plane is assumed to be the CAERO macro element plane.
4. The SPLINE EID is used only for error messages and need not be related to the macroelement identification number.

Input Data Entry SPOINT Scalar Point List

Description: Defines scalar points of the structural model

Format and Examples:

1	2	3	4	5	6	7	8	9	10
SPOINT	ID	ID	ID	ID	ID	ID	ID	ID	
SPOINT	3	18	1	4	16	2			

Alternate Form

SPOINT	ID1	"THRU"	ID2						
SPOINT	5	THRU	649						

Field

Contents

ID, ID1, ID2      Scalar point identification number (Integer > 0; ID1 < ID2)

Remarks:

1. If the alternate form is used, all scalar points ID1 through ID2 are defined.

Input Data Entry SUPPORT Fictitious Support

Description: Defines coordinates at which the user desires determinate reactions to be applied to a free body during analysis.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
SUPPORT	SETID	ID	C	ID	C	ID	C		
SUPPORT	1000	16	215						

Field

Contents

SETID                      Solution control SUPPORT set identification (Integer > 0).

ID                              Grid or scalar point identification number (Integer > 0).

C                                Component number (zero or blank for scalar points; any unique combination of the digits 1-6 for grid points).

Remarks:

1. Coordinates specified on this entry form members of a mutually exclusive set. They may not be specified on other entries that define mutually exclusive sets.
2. From one to eighteen support coordinates may be defined on a single entry.
3. Continuation entries are not allowed.

Input Data Entry TABDMP1      Modal Damping Table

Description:    Defines modal damping as a tabular function of frequency.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
TABDMP1	ID	TYPE	F <sub>1</sub>	G <sub>1</sub>	F <sub>2</sub>	G <sub>2</sub>	F <sub>3</sub>	G <sub>3</sub>	CONT
TABDMP1	3	G	0.0	0.005	1.0	0.008	2.0	0.001	ABC
CONT	F <sub>4</sub>	G <sub>4</sub>	F <sub>5</sub>	G <sub>5</sub>	F <sub>6</sub>	G <sub>6</sub>	etc		
+BC	2.5	.01057	2.6	.01362					

Field

Contents

ID                      Table identification number (Integer > 0).

TYPE                    Data word which indicates the type of damping units, "G", "CRIT", "Q", or blank. Default is "G".

F<sub>i</sub>                      Frequency value in cycles per unit time (Real ≥ 0.0).

G<sub>i</sub>                      Damping value (Real).

Remarks:

1. The F<sub>i</sub> must be in either ascending or descending order but not both.
2. Jumps between two points (F<sub>i</sub> - F<sub>i+1</sub>) are allowed, but not at the end points.
3. At least two entries must be present.
4. Any f<sub>i</sub>, g<sub>i</sub> entry may be ignored by placing the string "SKIP" in either of two fields used for that entry.
5. The TABDMP1 mnemonic infers the use of the algorithm

$$g = g_T(F)$$

where F is input to the table and g is returned. The table look-up, g<sub>T</sub>(F), is performed using linear interpolation within the table and linear extrapolation outside the table using the last two end points at the appropriate table end. At jump points the average g<sub>T</sub>(F) is used. There are no error returns from this table look-up procedure.

6. If TYPE is "G" or blank, the damping values are in structural damping units, that is, the value of g in (1+ig)K. If TYPE is "CRIT", the damping values are in the units of fraction of critical damping, C/C<sub>0</sub>. If TYPE is "Q", the

damping values are in the units of the amplification or quality factor,  $Q$ . These constants are related by the following equations:

$$C/C_0 = g/2,$$

$$Q = \begin{cases} 1/(2C/C_0), \\ 1/g. \end{cases}$$



### Input Data Entry TABLED1

**Description:** Defines a tabular function for use in generating frequency-dependent and time-dependent dynamic loads.

#### Format and Examples:

1	2	3	4	5	6	7	8	9	10
TABLED1	ID								CONT
TABDMP1	32								ABC
CONT	x <sub>1</sub>	y <sub>1</sub>	x <sub>2</sub>	y <sub>2</sub>	x <sub>3</sub>	y <sub>3</sub>	-etc-		CONT
+BC	-3.0	6.9	2.0	5.6	3.0	5.6			

#### Field

#### Contents

ID                      Table identification number (Integer > 0)

x<sub>i</sub>, y<sub>i</sub>                Tabular entries (Real)

#### Remarks:

1. The x<sub>i</sub> must be in either ascending or descending order but not both.
2. Jumps between two points (x<sub>i</sub> - x<sub>i+1</sub>) are allowed, but not at the end points.
3. At least two entries must be present.
4. Any x-y entry may be ignored by placing the string "SKIP" in either of the two fields used for that entry.
5. The generated function is:

$$Y = y_T(X)$$

where X is input to the table and Y is returned. The table look-up  $y_T(x)$  is performed using linear interpolation within the table and linear extrapolation outside the table using the last two end points at the appropriate table end. At jump points the average  $y_T(x)$  is used. There are no error returns from this table look-up procedure.

Input Data Entry TEMP Grid Point Temperature Field

Description: Defines temperature at grid points for determination of (1) Thermal Loading; and (2) data recovery.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
TEMP	SID	G	T	G	T	G	T		
TEMP	3	94	316.2	49	219.8				

Field

Contents

SID Temperature set identification number (Integer > 0).

G Grid point identification number (Integer > 0).

T Temperature (Real).

Remarks:

1. From one to three grid point temperatures may be defined on a single entry.
2. Average element temperatures are obtained as a simple average of the connecting grid point temperatures when no element temperature data are defined.
3. For each thermal load, temperatures must be specified for all grid points using either TEMP or TEMPD entries.

Input Data Entry TEMPD Grid Point Temperature Field Default

Description: Defines a temperature value for all grid points of the structural model which have not been given a temperature on a TEMP entry.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
TEMPD	SID	T	SID	T	SID	T	SID	T	
TEMPD	1	215.3							

Field

Contents

SID Temperature set identification number (Integer > 0 or blank).

T Default temperature value (Real).

Remarks:

1. From one to four default temperatures may be defined on a single entry.
2. Average element temperatures are obtained as a simple average of the connecting grid point temperatures when no element temperature data are defined.
3. For each thermal load, temperatures must be specified for all grid points using either TEMP or TEMPD entries.

# Input Data Entry TF Dynamic Transfer Function

## Description:

1. Used to define a transfer function of the form

$$(B + B_1p + B_2p^2)u_d + \sum_i (A_0(i) + A_1(i)p + A_2(i)p^2)u_i = 0.0$$

2. May also be used as a means of direct matrix input. See Remark 3.

## Format and Examples:

1	2	3	4	5	6	7	8	9	10
TF	SID	GD	CD	BO	B1	B2			CONT
TF	1	2	3	4.0	5.0	6.0			+ABC
CONT	G(1)	C(1)	A0(1)	A1(1)	A2(1)				CONT
+ABC	3	4	5.0	6.0	7.0				

## Field

## Contents

SID	Set identification (Integer > 0).
GD,G(i)	Grid, scalar or extra point identification numbers (Integer > 0).
CD,C(i)	Component numbers (null or zero for scalar or extra points, any <u>one</u> of the digits 1-6 for a grid point).
BO,B1,B2 A0(i),A1(i), A2(i)	Transfer function coefficients (Real).

## Remarks:

1. The matrix elements defined by this entry are added to the dynamic matrices for the problem.
2. Transfer function sets must be selected in Solution Control (TFL = SID) to be used.
3. The constraint relation given in the equation will hold only if no structural elements or other matrix elements are connected to the dependent coordinate,  $u_d$ . In fact, the terms on the left side of the equation are simply added to the terms from all other sources in the row for  $u_d$ .
4. Any number of continuations are allowed.

### Input Data Entry TIMELIST

Description: Defines a list of times at which outputs are desired.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
<u>TIMELIST</u>	<u>SID</u>	<u>TIME</u>	<u>TIME</u>	<u>TIME</u>	<u>TIME</u>	<u>TIME</u>	<u>TIME</u>	<u>TIME</u>	<u>CONT</u>
<u>TIMELIST</u>	100	0.1	0.2	0.5	1.0				
<u>CONT</u>	<u>TIME</u>	<u>TIME</u>	-etc-						

Field

Contents

**SID** Set identification number referenced by Solution Control  
Integer > 0 )

**TIME** Time, (in consistent time unit) at which outputs are de-  
sired. (Real)

Remarks:

1. In order to be used, the SID must be referenced by Solution Control.
2. The nearest time to TIME, either above or below, which was used in the Transient Response analysis will be used to satisfy the output requests.
3. Any number of continuations is allowed.

Input Data Entry TLOAD1

Description: Defines a time dependent motion of the form:

$$P(T) = (A F (t - r))$$

for use in a transient response problem.

Format and Examples:

1	2	3	4	5	6	7	8	9	10
TLOAD1	SID	DLAGID	TID						
TLOAD1	10	8	13						

Field

Contents

SID                      Set identification number (Integer > 0)

DLAGID                  Identification number of DLAGS set which defines A and r  
(Integer > 0)

TID                      Identification number of a TABLED1 entry which gives F(t-r)  
(Integer > 0)

Remarks:

1. SID must be unique for all TLOAD1, TLOAD2, RLOAD1, and RLOAD2 entries.

## Input Data Entry TLOAD2

**Description:** Defines a time-dependent dynamic load of the form:

$$(P(t)) = \begin{cases} (0), & \bar{t} < 0 \text{ or } \bar{t} > T2 - T1 \\ (A\bar{t}^B e^{C\bar{t}} \cos(2\pi F\bar{t} + \phi)), & 0 \leq \bar{t} \leq T2 - T1 \end{cases}$$

for use in a transient response problem where  $\bar{t} = t - T1 - \tau$ .

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
TLOAD2	SID	DLAGID	T1	T2	FREQ	PHASE	CTEXP	GROWTH	
TLOAD2	10	6	2.1	4.7	12.0	30.0	2.0	3.0	

### Field

### Contents

SID	Set identification number (Integer > 0)
DLAGID	Identification number of the DLAGS entry set which define the time invariant load A and the time delay $\tau$ (Integer > 0)
T1	Time constant (Real $\geq 0.0$ )
T2	Time constant (Real, $T2 > T1$ )
FREQ	Frequency in cycles per unit time (Real $\geq 0.0$ )
PHASE	Phase angle in degrees (Real)
CTEXP	Exponential coefficient (Real)
GROWTH	Growth coefficient (Real)

### Remarks:

1. TLOAD2 loads may be combined with TLOAD1 loads only by specification on a DLOAD entry.
2. SID must be unique for all TLOAD1, TLOAD2, RLOAD1 and RLOAD2 entries.

# Input Data Entry TRIM Trim Variable Constraint

Description: Specifies conditions for aeroelastic trim analysis.

## Format and Example:

1	2	3	4	5	6	7	8	9	10
TRIM	TID	MACH	QDP	SYMXZ	TRMTYP	NZ	QRATE	VO	
TRIM	1	.9	100.	1	1	1.0	0.0	0.0	

## Field

## Contents

TID	Trim set identification number (Integer > 0)
MACH	Mach number (Real > 0.0)
QDP	Dynamic pressure (Real > 0.0)
SYMXZ	Symmetry key for aero coordinate xz plane (Integer) (+1 for symmetry, 0 for no symmetry, -1 for antisymmetry).
TRMTYP	Type of trim required (0 = No trim, 1 = trim lift forces only, 2 = trim lift and pitching moment)(Integer)
NZ	Load factor or acceleration (Real)
QRATE	Aircraft pitch rate (rad/sec)(Real)
VO	Aircraft velocity (Real)

## Remarks:

1. The TRIM entry is selected in solution control by "TRIM = TID."
2. Units on the inputs are:

QDP - Force/unit area.

NZ - This input is dimensioned with units of length/sec<sup>2</sup> unless a MASS conversion factor has been given, in which case NZ is non-dimensional. Acceleration used by the program is equal to NZ/MASS, where MASS is input on the CONVERT data entry or is defaulted to 1.0.

QRATE - Rad/sec

$$q_{rate} = \frac{g(NZ-1.0)}{VO}$$

VO - Length/sec

where the length, area and force units must be consistent with the remaining bulk data entries.



3. QRATE and VO are required only when TRMTYP - 2.
4. Symmetric analyses are for longitudinal motions while anti-symmetric analyses are for lateral motions.

## Input Data Entry TSTEP

**Description:** Defines time step intervals at which a solution will be generated and output in transient analyses.

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
TSTEP	SID	N(1)	DT(1)	NO(1)					CONT
TSTEP	2	10	.001	5					+ABC
CONT		N(2)	DT(2)	NO(2)					CONT
+ABC		9	0.01	1					

-etc.-

### Field

### Contents

SID	Set identification number (Integer > 0)
N(i)	Number of time steps of value DT(i) (Integer $\geq 2$ )
DI(i)	Time increment (Real > 0.0)
NO(i)	Skip factor for output (every NO(i) <sup>th</sup> step will be saved for output) (Integer > 0)

### Remarks:

1. TSTEP entries must be selected in the Solution Control (TSTEP-SID).
2. Note that the entry permits changes in the size of the time step during the course of the solution. Thus, in the example shown, there are 10 time steps of value .001 followed by 9 time steps of value .01. Also, the user has requested that output be recorded for  $t = 0.0, .005, .01, .02, .03$ , etc.

## Input Data Entry VSDAMP

**Description:** Specifies values of  $g$  and/or  $\omega_3$  to generate either viscous damping that has the same damping forces as structural damping of magnitude  $g$  at the frequency  $\omega_3$  or to specify the structural damping  $g$  (see Remarks 3 and 4)

### Format and Examples:

1	2	3	4	5	6	7	8	9	10
VSDAMP	SID	G	$\omega_3$	SID	G	$\omega_3$			
VSDAMP	100	0.005	15.0						

### Field

### Contents

SID                      Set identification number (Integer > 0)

G                        Damping value (Real)

$\omega_3$                       Frequency value in Hertz (Real)

### Remarks:

1. The setid is selected by the DAMPING-n command in Solution Control.
2. Up to two values of  $g$  and  $\omega_3$  can be defined on a single entry.
3. If  $\omega_3$  is zero,  $g$  will be used to generate a complex stiffness matrix of the form  $K = (1 + ig)K$ .
4. If  $\omega_3$  is nonzero, a viscous damping matrix of the form

$$[B] = \frac{g}{2\pi\omega_3} [K]$$

is generated.